

ترجمة المهندس :محمد على يوسف

الطبعة الأولى ٢٠٠٦



تأليف : اندرو فوس

إعداد و ترجمة المهندس : محمد على يوسف

عزيزى القارئ

بسم الله والصلاة والسلام على اشرف المرسلين سيدنا محمد صلى الله علية وسلم وعلى اله وصحبة أجماعيين

أما بعد

أردت أنى أقوم بعملية ترجمة بعد الكتب الالكترونية التي تخص بمجال الحاسبات الآلية مثل (كتب لغات البرمجة – وأنظمة التشغيل – والشبكات) وكل ما يخص بالحاسب الالى

لقد تمت عملية ترجمة هذا الكتاب وهو يتحدث في (مقدمة فيجوال بيسك دوت نت ٢٠٠٥ بواسطة اندرو فوس)

الذي قام بتأليفه الكاتب اندرو فوس والذي أعطى له اسم
" Introduction to Programming with Visual Basic.Net 2005 by Andrew Vos "

وهذا رابط الكتاب عن طريق الانترنت للإطلاع على النسخة الأصلية باللغة الإنجليزية
http://books.wickedorange.com/vb2005intro/book/start/start.html

ملحوظة لقارئ الكتاب

- أنا ليس محترف في بلاغة اللغة الإنجليزية
- عند وجود خطأ ما لا تتردد في كتابة رسالة صغيرة ترسلها على هذا الميل ensert.net2000@gmail.com لتغيير الخطأ وإعادة نشر الكتاب مرة أخرى بعد التصحيح .

نوجوا أن يحوز الكتاب قبول القارئ العربي وان يحقق النفع المرجو منة والله ولى التوفيق

المهندس : محمد على يوسف

مقدمة فيجوال بيسك دوت نت ٢٠٠٥

أننا نعيش في عالم متغير باستمرار من التكنولوجيا والاختراع. الشيء الوحيد الذي تغير هو بعض هذه الأيام.

الابتكارات الحديثة التي أوصلتنا الهواتف الخلوية ، وعالية السرعة بشبكه الانترنت ، والتصوير الرقمي وغيرها الكثير من الأفكار الثورية. المشترك رابط بين هذه الأفكار هو انه ، في مكان ما على طول الخط ، مبرمج كتب البرمجيات لجعلها أكثر من مجرد فكرة ، التي قطعتها هذه الأفكار موجودة ، وبدونها سنكون في عالم مختلف جدا.

في هذا الكتاب أني اعتزم البدء لكم من على طريقك إلى معرفة واسعة ومتشعبة فن البرمجة.

عن المؤلف

اندرو فوس ولدت في كيب تاون ، جنوب أفريقيا.

الذين ينبغي قراءة هذا الكتاب؟

أي شخص مهتم في تعلم أساسيات كيفية تطوير البرمجيات وينبغي أن يقرأ هذا الكتاب. ومن المعتزم للناس الذين يعرفون شيئا البتة عن البرمجة ، ولكن معرفة كيفية استخدام الحاسوب.

ما هي البرمجيات هل أنت بحاجة؟

مايكروسوفت فيجوال بيسك ٢٠٠٥ اكسبريس أو مايكروسوفت فيجوال استديو ٢٠٠٥) هي التي استخدمت لصنع البرامج أو تجميع مدونة الامثله التي وردت في هذا الكتاب ، على الرغم من أنه ليس مطلوبا.

الاتفاقيات المستخدمة في هذا الكتاب

قائمة الملاحة

A → رمز يستخدم لتمثيل قائمة الانفصال ، وعلى سبيل المثال عندما كنت أنظر مساعدة → عنك ينبغي انقر على قائمة مساعدة وثم انقر على قائمة حول هذا البند.

شفره المصدر

شفره المصدر يرد في مختلف وحجم الخط لسهولة القراءة.

```
Public Sub New()
   Me.DoStuff()
End Sub
```

الروابط

Microsoft Developers Network
http://www.msdn.net

Microsoft Developers Network 2 Library
http://msdn2.microsoft.com

CodeProject
http://www.codeproject.com

ما هي البرمجة؟

البرمجة هو في جوهره code إنشاء التعليمات أن أقول حاسوب أو آلة ما نفعله.

ما هي لغة برمجة؟

لغة برمجة ما هو استخدام المبرجين لإنشاء البرمجيات. فهو يجعل الحياة أسهل بكثير بالنسبة لنا المبرجين عن طريق السماح لنا لتطوير البرمجيات في حقوق قراءتها ، من السهل فهم هذه اللغة.

برجيات هو "Codes" في لغة من اختيارك وعندئذ يحصل على جمعها في ملف ثنائي بحيث تكون أداة يمكن فهمه. وهذا ما يسمى "Machine Code" ، والذي يشار إليه أيضا باسم ألقوميه Machine Code. جمع هذا Code هو تحويل حقوق مقروء مدونة مكتوب في لغة برمجة Machine لغة الآلة.

وفي حالة microsoft.net ، شفره المصدر الخاص بك هو تجميع للغة وسيطة (IL) ، الذي يختلف عن Machine Code. Msil هو المعروف أيضا باسم مايكروسوفت لغة وسيطة (msil) أو اللغة المشتركة وسيطة (Msil .(cil هو واضح في بعض الأحيان "Missile".

لغة وسيطة هي لغة جديدة والذي صمم ليكون بكفاءة تحويلها إلى آلة ألمدونه على أنواع مختلفة من الاجهزه. جميع لغات microsoft.net توليد لغة وسيطة خلال تجميع الملفات.

لأن هذه المدونة لغة وسيطة هي مختلفة عن القومية لغة الآلة وهي تتطلب شيء يسمى اللغة المشتركة وقت التشغيل (clr) ليتم تثبيته على جهازك لتشغيلها. اللغة المشتركة وقت التشغيل هو المكون الأساسي لل microsoft.net الإطار أن ينفذ ايل code ، ودون أن حسابك تجميعها. الصافية الملفات سيكون عديم الفائدة. Clr فإن لديه القدرة على أن تعمل بشكل مختلف على مختلف الاجهزه ، على سبيل المثال باستخدام بيئات متعدد المعالج من اجل الحصول على المزيد من القوة المتعددة للوحدة المعالجة المركزية.

ما هو NET Framework. ؟

فإن - **NET Framework.** هو تطوير وتنفيذ بيئة تتيح لغات البرمجة المختلفة إلى العمل معا من اجل تهيئة ويندوز التطبيقات القائمة.

ببساطة أكثر ، وهي وسيلة لخلق برامج ، وسيلة لتنفيذ البرامج.

إلى استخدام برامج أن تتم. الصافية ، والحاجة إلى المستخدمين لديها . Net Framework وهو متاح من تحديث ويندوز.

تضمن NET Framework.

The Common Language Runtime (CLR)

فان clr هي لغة محايدة التنمية & إعدام البيئة. وهي توفر أيضا خدمات للمساعدة في إدارة تطبيق عقوبة الإعدام.

The Framework Class Libraries (FCL)

في إطار الطبقة مكتبات تحتوي مجموعة متنوعة من المفيد الوظيفة أن نأخذها بعين الاعتبار في برامجنا ، وتقديم المساعدة اللازمة لتطوير البرامج التي هي أكثر استقرارا ، وتسريع عملية التنمية.

دعم لبروتوكولات الشبكة الموحدة والمواصفات

فإن - صافي الإطار يشمل دعم بروتوكولات الانترنت القياسية ومواصفات مثل مشاريع التعاون التقني / الملكية الفكرية ، والصابون ، وأكس أم أل ، & بروتوكول انتقال النص المتشعب.

دعم لغات البرمجة المختلفة

تطوير البرمجيات ل. **NET Framework** الذي يمكن عمله في مجموعة متنوعة من مختلف لغات البرمجة ، بحيث لك اختيار اللغة التي تناسب أفضل لكم.

فرق للمطورين العمل على المشروع نفسه يمكن أيضا استخدام مختلف لغات البرمجة في نفس الوقت الذي نتائج أكثر إنتاجيه في بيئة العمل.

الدعم لمختلف المنابر

فإن - **NET Framework** هو متاح لكثير من منصات ويندوز المختلفة ، على سبيل المثال ويندوز اكس بي (الحاسوب الشخصي المكتبي أو الحواسب الحجرية (Cell Phones والهواتف الخلوية وغيرها) etc)

ما هي المشاريع التي يمكن أن أقوم بإنشائها؟

العميل الذكى تطبيقات ويندوز

نوافذ التطبيقات الأساسية التي لها واجهة المستخدم الرسوميه (واجهة المستخدم الرسوميه) هي عادة أول ما ينبغي أن المبرمجين مهتمون خلق.

الذكية العملاء على تطبيقات ويندوز مع المزيد من الميزات المتقدمة ، مثل القدرة على استخدام خدمات الشبكة العالمية (RSS Feed Readers) وتسمح برمجيات التحديثات التلقائية. مع. Net. ومن الأسهل بكثير أن إنشاء التطبيقات الذكية العملاء أكثر من أي وقت مضى!

التطبيقات الشبكية

فإن The .NET Framework يتضمن مجموعة من السمات القوية على الانترنت ، المعروفة باسم asp.net ، التي تسمح لك لإنشاء المشاريع الصغيرة ضخمة كالتطبيقات الشبكية.

خدمات الشبكة

خدمات الشبكة العالمية هي وسيلة لتبادل الرسائل بين نظامين. مع asp.net يمكنك إنشاء خدمات ويب XML إن الرد على رسائل تنقل أكثر من بروتوكول انتقال النص المتشعب.

نوافذ الخدمات

ويندوز خدمات التطبيقات التي تديرها في خلفية أداء المهام. فهي تسيطر عليها مدير دائرة الرقابة. إيجاد خدمات ويندوز تستخدم لتكون أصعب بكثير ولكن مع - صافي الإطار ومن بالسهولة خلق أي نوع آخر من التطبيق.

تعزية التطبيقات

تعزية التطبيقات هي جميلة كثير نفس القيادة مدونة في ويندوز (انقر → Start البعيد ، من نوع "cmd.exe" والصحافة طيب أن نرى قيادة مدونة).
في. net يمكننا بسهولة إيجاد تعزية التطبيقات.

الطبقة مكتبات

A الطبقة المكتبة هي أساسا مكتبه المدونة. ويمكننا تخزين أقسام المدونة داخل الطبقة المكتبة وربط مشاريعنا إلى انه بدلا من إعادة تشفير نفس الشفرة في كل المشاريع التي تستخدم فيها. فهي تتيح لنا أساسا لإعادة استخدامها عام أن المدونة هي شائعة الاستعمال حتى إننا ابدأ إلى code جديد.

A الطبقة التي أنشئت في المكتبة net. يمكن استخدامها في أي مكان آخرnet.المشروع

جهاز التطبيقات الذكية

كما net. A المطور يمكننا أيضا أن تكتب التطبيقات التي تستهدف . NET Compact Framework ، وهو compacted نسخة لل. NET Framework، الذي هو الأمثل ليتم

تشغيله على الاجهزه بموارد اقل من عادي الحاسوب ، على سبيل المثال الاجهزه اليدوية. Visual Studio 2005 وقد ذكية جهاز المحاكي للمساعدة على تطوير التطبيقات الذكية جهاز واحد دون فعلا امتلاك أداة ذكية.

إعداد ونشر المشاريع

مع Visual Studio 2005 يمكننا أن نخلق المثبت مجموعات قوية لمشاريعنا. وإذا استخدمنا إيجاد clickonce المثبت ثم يمكن أن يكون لنا برنامجنا دوري التحقق من التحديث بدون كتابة أي code!

الجزء الأول
المفاهيم الأساسية

الفصل الأول

المتغيرات

المتغيرات هي من المحتمل جدا الأكثر استخداما هي سمة في إي لغة برمجة ، وأنها هي التي نستخدمها لتخزين البيانات فيها.

متغير له اسم ، ونوع البيانات ، وقيمة.

الاسم : الاسم هو ما نستخدمه للوصول إلى هذا المتغير ، والخروج منها أو مجموعة بياناته.
نوع البيانات : بيانات النوع هو نوع البيانات التي ستخزن في المتغير ، على سبيل المثال صورة سوف تخزن معلومات عن الألوان ومواقع للبكسل في صورة ، والتاريخ سوف تخزن معلومات عن اليوم والشهر والسنة .
القيمة : القيمة الفعلية البيانات المخزنة في المتغير.

النظر في المثال التالي.

```
Dim TextInfo As String = "This is some data"
```

أساسا ما قطعه من هذه المدونة هل هو خلق جديد متغير يسمى "textinfo" مع نوع البيانات "String" ، وهو نوع البيانات Visual Basicالتي نص مخازن المعلومات ، وتحدد قيمته إلى "This is some data". عندما خلق سلسلة القيمة ونحن أرفق البيانات في الأقوال الماثوره. "Dim" هي الكلمات الرئيسية المستخدمة في النص التشكيلي الأساسية للإعلان عن متغير ، ولكننا سوف أعرفك المزيد عن ذلك في وقت لاحق.

الفصل الثاني

أنواع البيانات المشتركة

Boolean

تستخدم لتخزين حقيقية أو زائفة القيمة.

```
Dim DotNetIsEasy As Boolean = True
```

Byte

تستخدم البيانات التي تحتوي على ثنائي ، تتراوح بين ٠ إلى ٢٥٥.

```
Dim SingleByte As Byte = 120
```

Char

تستخدم لتخزين واحدة الطابع.

```
Dim CharValue As Char = "C"
```

Date

تستخدم لتخزين موعد القيمة.

```
Dim DateValue As Date = DateTime.Today
```

DateTime.Today يحصل التاريخ الحالي

Decimal

تستخدم لتخزين القيمة العشرية.
اكبر قيمة ممكنة هو + / -٧٩٢٢٨١٦٢٥١٤٢٦٤٣٣٧٥٩٣٥٤٣٩٥٠٣٣٥ مع ٢٨ أرقام عشريه.

```
Dim DecimalValue As Decimal = 2.0001232
```

Double

تستخدم لتخزين مزدوجة الدقة النقطة العائمة الأرقام أن يراوح في قيمة من - ١,٧٩٧٦٩٣١٣٤٨٦٢٣١٥٧٠ e +308 طريق - ٤,٩٤٠٦٥٦٤٥٨٤١٢٤٦٥٤٤ e - 324 للقيم السلبية ومن ٤,٩٤٠٦٥٦٤٥٨٤١٢٤٦٥٤٤ e - 324 عن طريق ١,٧٩٧٦٩٣١٣٤٨٦٢٣١٥٧٠ e +308 القيم الايجابية.

```
Dim DoubleValue As Double = 2.1221
```

Integer

عدد صحيح يستخدم لتخزين القيم التي يمكن أن تتراوح بين -٢١٤٧٤٨٣٦٤٨ إلى ٢١٤٧٤٨٣٦٤٧.

```
Dim LongValue As Integer = 23
```

Long

عدد صحيح يستخدم لتخزين القيم التي يمكن ان تتراوح بين -٩٢٢٣٣٧٢٠٣٦٨٥٤٧٧٥٨٠٨ ل٩٢٢٣٣٧٢٠٣٦٨٥٤٧٧٥٨٠٧.

```
Dim LongValue As Long = 23
```

Object

تستخدم لتخزين وصلات إلى أي نوع من الاعتراض.

```
Dim ObjectValue As Object = "We will store a String here"
```

Short

عدد صحيح يستخدم لتخزين القيم التي تتراوح بين -٣٢٧٦٨ إلى ٣٢٧٦٧.

```
Dim ShortValue As Short = 23
```

Single

تستخدم لتخزين واحدة الدقة النقطة العائمة أرقام تتراوح قيمتها من -٣,٤٠٢٨٢٣٥ e +38 طريق - ١,٤٠١٢٩٨ e - 45 للقيم السلبية ومن ١,٤٠١٢٩٨ e - 45 عن طريق ٣,٤٠٢٨٢٣٥ e +38 للقيم الايجابية. أحادي الدقة تخزين أرقام تقريبية من العدد الحقيقي.

```
Dim SingleValue As Single = 0.1
```

String

تستخدم لتخزين سلسلة من يصل إلى ما يقرب من ٢ مليار (٢ ^ ٣١) حرفا.

```
Dim StringValue As String = "This is a string"
```

الفصل الثالث

المعاملات

المعاملات الحسابية

المعامل ^

يثير عددا من السلطة إلى عدد آخر.

```
result = number ^ exponent
```

والنتيجة هي عدد المثارة إلى السلطة من داعية ، وكما دائما مزدوجة القيمة. سواء من حيث العدد وداعية يجب ان يكون من النوع المزدوج البيانات ، أو خلاف ذلك سيحصل تلقائيا تحويلها إلى مضاعفة. قيمة للداعية يمكن جزئي ، سلبية ، أو كليهما.

عندما يكون هناك أكثر من واحد الترقية أدوا تعبيرا واحدا ، هو معامل ^ تقييمها كما هي واجهت من اليسار إلى اليمين.

معامل *

يضاعف الرقمين.

```
result = number1 * number2
```

والنتيجة هي نتاج number1 و number2.
جميع أنواع عدديه تحظى بدعم والبيانات الناتجة من النوع تتوقف على نوع من يطبق عليهم العملية الرياضية.

المعامل /

يقسم الرقمين والعودة عائم نقطة نتيجة.

```
result = number1 / number2
```

جميع أنواع عدديه هي مدعومة. والنتيجة هي القسمة على السواء بما في ذلك أي أرقام المتبقية. البيانات الناتجة يتوقف على نوع من أنواع البيانات الرقمين.

**المعامل **

يقسم الرقمين والعودة نتيجة وجود عدد صحيح.

```
result = number1 \ number2
```

جميع أنواع عدديه هي مدعومة. والنتيجة هي خارج القسمة عدد صحيح من number1 مقسوما number2 ، التي تزيل أي تبقى ويبقى سوى جزء عدد صحيح. البيانات الناتجة يتوقف على نوع من أنواع البيانات الرقمين.

المعامل Mod

يقسم الرقمين والعودة فقط الباقي.

```
result = number1 Mod number2
```

جميع أنواع عدديه هي مدعومة. والنتيجة هي ما تبقى بعد خروج number2 ينقسم number1.

معامل +

ويضيف الرقمين معا.

```
result = number1 + number2
```

النتيجة هي مجموع كل الأرقام. أي عددي أنواع مدعمه.

المعامل -

العودة الفرق بين اثنين من التعبيرات الرقمية.

```
result = number1 - number2
```

أي قيم رقمية يتم دعم. والنتيجة هي الفرق بين number1 و number2.

إحالة معاملي

المعامل =

يسند إلي قيمة متغير.

```
result = variable
```

أي أنواع البيانات هي مدعومة.

المعامل = ^

يرفع قيمة متغير إلى السلطة للتعبير عن ويسند نتيجة عودة إلى المتغير أو الممتلكات.

```
variable ^= expression
```

أي بيانات رقمية هي أنواع الدعم.

المعامل = /

يقسم قيمة متغير به للتعبير عن القيمة.

```
result /= expression
```

وتؤيد أي نوع بيانات رقمية.

**المعامل = **

يقسم قيمة متغير به للتعبير عن القيمة.

```
result \= expression
```

وتؤيد أي نوع بيانات رقمية.

المعامل = +

ويضيف قيمة إلى عدد قيمة عدد آخر.

```
result += expression
```

النتيجة هي مجموع كل الأرقام. وتؤيد أي نوع بيانات رقمية.

المعامل =-

يطرح قيمة عدد من قيمة أخر عدد.

```
result -= expression
```

والنتيجة هي تعبير عن حق تطرح من التعبير عن اليسار. وتؤيد أي نوع بيانات رقمية.

<<= معامل

يؤدي الحساب على اليسار تحول قيمة متغير ويسند نتيجة عودة إلى المتغير.

```
expression1 <<= expression2
```

<< = المعامل

يؤدي التحول على حساب الحق قيمة متغير ويسند نتيجة عودة إلى المتغير.

```
expression1 >>= expression2
```

& = المعامل

يسلسل سلسلة المتغير لسلسلة متغيرة.

```
string1 >>= string2
```

بت التحول معاملي

<<المعامل

يؤدي التحول على حساب اليسار قليلا النمط.

```
result = pattern << amount
```

وستكون النتيجة نتائج تحويل نمط قليلا من المبلغ. نوع البيانات عاد سوف يكون نفس النمط.

<< المعامل

يؤدي التحول على حساب الحق قليلا النمط.

```
result = pattern >> amount
```

وستكون النتيجة نتائج تحويل نمط قليلا من المبلغ. نوع البيانات عاد سوف يكون نفس النمط.

مقارنة بين المعاملين

المعامل <

ويقارن اثنين قيم رقمية لتحديد ما إذا كانت واحدة اقل من الأخرى.

```
result = number1 < number2
```

ونتيجة ذلك هي تحديد قيمة منطقي إذا number1 هو اقل من number2.

معامل <=

ويقارن اثنين قيم رقمية لتحديد ما إذا كان احد أقل من أو مساو لغيره.

```
result = number1 <= number2
```

ونتيجة ذلك هي تحديد قيمة منطقي إذا number1 هو أقل من أو يساوي number2.

المعامل >

ويقارن اثنين قيم رقمية لتحديد ما إذا كانت واحدة اكبر من الأخرى

```
result = number1 > number2
```

ونتيجة ذلك هي تحديد قيمة منطقي إذا number1 اكبر من number2.

المعامل > =

ويقارن اثنين قيم رقمية لتحديد ما إذا كان احد اكبر من أو مساو لغيره.

```
result = number1 >= number2
```

ونتيجة ذلك هي تحديد قيمة منطقي إذا number1 هو اكبر من أو يساوي number2.

= المعامل

يقارن بين اثنين من المتغيرات وتحديد ما إذا كان أنهم متساوون.

```
result = variable1 = variable2
```

وتعيد تحديد قيمة منطقي إذا variable1 يساوي variable2.

<> معامل

يقارن بين اثنين من المتغيرات وتحديد ما إذا كان أنهم ليسوا على قدم المساواة.

```
result = variable1 <> variable2
```

وتعيد تحديد قيمة منطقي إذا variable1 مختلفا عن variable2.

is المعامل

يقارن بين اثنين من وجوه إشارة المتغيرات.

```
result = objectReference1 Is objectReference2
```

ونتيجة ذلك هي تحديد قيمة منطقي إذا اثنين وجوه إشارة إلى المتغيرات نقطة الجسم ذاته.

Isnot المعامل

يقارن بين اثنين من وجوه إشارة المتغيرات.

```
result = objectReference1 Is objectReference2
```

ونتيجة ذلك هي تحديد قيمة منطقي إذا اثنين وجوه الإشارة إلى نقطة المتغيرات بمختلف الأشياء.

Like المعامل

ويقارن سلسلة ضد نمطا معينا.

```
result = stringValue Like pattern
```

والنتيجة هي قيمة منطقي يشير إلى ما إذا كان string أو لا تستوفي النمط. إذا كان كل من string ونمط تعتبر لاغيه بعد ذلك نتيجة صحيح.

سلسلة معاملي

المعامل &

يولد سلسلة من التعبيرات اثنين.

```
result = stringValue1 & stringValue2
```

والنتيجة هي سلسلة قيمة stringvalue1 وstringvalue2 انضمت معا.

معامل +

يولد سلسلة سلاسل من التعبيرات اثنين.

```
result = stringValue1 + stringValue2
```

والنتيجة هي سلسلة قيمة stringvalue1 وstringvalue2 انضمت معا.

Logical/Bitwise Operators

المعامل And

يؤدي منطقيا إلى جانب عبارات منطقي على اثنين ، أو bitwise جنبا إلى جنب على اثنين من التعبيرات الرقمية.

```
result = expression1 And expression2
```

منطقي للمقارنة ستكون النتيجة المنطقية جنبا إلى جنب من اثنين منطقي القيم.
وستكون النتيجة إذا كان صحيحا سواء expression1 وexpression2 تقييم صحيح ل.

لbitwise عمليات النتيجة هي قيمة عدديه تمثل bitwise بالاشتراك اثنين من أنماط عدديه قليلا.

المعامل Not

يؤدي منطقيا إنكار منطقي على التعبير ، أو bitwise الإنكار على رقمية التعبير.

```
result = Not expression
```

وإذا كان التعبير يقيم إلى صحيحا ثم ستكون النتيجة كاذبة ، والعكس بالعكس.

المعامل OR

يؤدي منطقيا المفرق على اثنين منطقي عبارات ، أو bitwise المفرق على اثنين من التعبيرات الرقمية.

```
result = expression1 Or expression2
```

منطقي للمقارنة النتيجة ستكون شاملة منطقي المفرق اثنين منطقي القيم.
لbitwise العمليات ستكون النتيجة قيمة عدديه تمثل شاملة bitwise المفرق اثنين من أنماط عدديه قليلا.

معامل XOR

يؤدي منطقيا استبعاد اثنين على عبارات منطقي ، أو bitwise الاستبعاد على اثنين من التعبيرات الرقمية.

```
result = expression1 Xor expression2
```

منطقي للمقارنة ستكون النتيجة المنطقية استبعاد اثنين منطقي القيم. لbitwise العمليات ستكون النتيجة قيمة عدديه تمثل bitwise استبعاد اثنين من أنماط عدديه قليلا.

المعامل Andalso

يؤدي قصيرة الدوران المنطقية جنبا إلى جنب على اثنين من التعبيرات.

```
result = expression1 AndAlso expression2
```

والنتيجة هي منطقي نتيجة للمقارنة بين اثنين من التعبيرات.

المعامل Orelse

يؤدي قصيرة الدوران شامل منطقي المفرق على اثنين من التعبيرات.

```
result = expression1 OrElse expression2
```

والنتيجة هي قيمة منطقي. إذا كان لأحدها أو لكليهما لتقييم التعبيرات الحقيقية ، نتيجة صحيح. والجدول التالي يوضح كيفية نتيجة عاقدة العزم.

معاملات متنوعة

المعامل Addressof

يخلق الأجراء مندوب المثال إن الإشارات المحددة الأجراء.

```
AddressOf procedureName
```

في هذه المدونة procedurename هو الأجراء الذي سيتم المشار إليها من قبل مندوب الأجراء الذي أنشئ حديثا. عندما يحتج مندوب المحدد أسلوب يسمى. مزيد من المعلومات حول المندوبين سيتم تغطيتها في وقت لاحق.

المعامل Gettype

عودة نوع الجسم لنوع محدد.

```
GetType(TypeName)
```

عودة نوع الجسم للمحدد typename. يمكنك تمرير اسم أي نوع البصرية البيانات الأساسية النوع أو هذا السرد ، ونوع إي اسم الطبقة ، والمندوب ، والسرد ، وحده ، أو وصلة الهيكل المحدد في طلبك ، أو نوع من أي اسم. الصافية في إطار الطبقة ، تعداد ، وحده ، وصلة أو هيكل.

Gettype هو مفيد إذا كنت بحاجة إلى معرفة ما اذا كان اثنان الأشياء هي من نفس النوع.

المعامل Typeof

ويقارن جسم الإشارة إلى متغير نوع البيانات.

```
result = TypeOf variable Is Type
```

A منطقي قيمة العائدات ، وتحديد ما إذا كانت من نوع المتغير هو النوع.

النظر في المثال التالي.

```
result = TypeOf "StringValue" Is String
```

النتيجة ستكون متساوية صحيح ، لأن المتغير تجاوزنا هو نوع من String

الفصل الرابع

وصول معدلات

ما هى "الوصول إلى المعدلات"؟

وصول معدلات هي الكلمات الرئيسية التي تتحكم في الوصول إلى مستوى العناصر في المدونة.

أقول لدينا اثنين من الأجسام ، object1 وobject2. إذا object1 له طريقة تسمى dostuff إن يعلن الخاص ، وعندئذ فقط المدونة داخل هذا الجسم سوف يسمح للدعوة ، أو حتى انظر هذه الطريقة. إذا dostuff هو Public ، ثم أعلن أن أي جسم سوف يسمح لهم انظر والذي يطلق عليه.

اعتقد من ضوء التحول كما جسم أو الطبقة. الفعلي زر إننا الصحافة ستكون عامة ، متاحة للجميع ، ولكن داخل الأسلاك هو الخاص ، لأنه ليس المطلوب منا إن تؤديه حولها مع الأسلاك للحصول على ضوء العمل ؛ ونحن للتو قد تصل إلى هذا التحول.
وصول معدلات تستخدم مثل هذا ، عندما تصنع الجسم التي قد مهام محددة متاحة للجميع ، وغيرها من المهام التي تتحكم في أعمال الداخلية للجسم ، التي لا تتوفر إلا لمدونة داخل ذلك الجسم.

Public

تحديد Public يسمح عنصر في السؤال الذي يتعين الوصول إليه في أي مكان من مدونة

```
Public element As Element
```

يمكنك استخدام Public فقط في التفاعل ، وحده ، أو على مستوى المجرد.

Protected

تحديد Protected تتيح للعنصر في السؤال الذي يتعين الوصول أليه من مدونة إلا في هذا المستوى ، أو في أي sublevels.

```
Protected Class SampleClass
```

يمكنك استخدام Protected فقط على مستوى الطبقة عندما يعلن احد أفراد وحدة الطبقة

Friend

صديق تتيح تحديد عنصر في المسالة التي ينبغي الوصول إليها إلا من داخل المدونة الحالية للجمعية العامة ,(application or class library)، وليس من أي مكان خارج الجمعية.

```
Friend element As Element
```

يمكنك استخدام وصلة فى صديق فقط ، وحده أو على مستوى المجرد.

Protected Friend

تحديد Protected والصديق معا يسمح عنصر في المسالة إلى إن يتم الوصول إلى الإطلاع فقط من قانون مستمد من داخل الصفوف أو داخل الجمعية نفسها.

```
Protected Friend element As Element
```

يمكنك استخدام Protected صديق فقط على مستوى الطبقة.

Private

تحديد Private ويسمح عنصر في المسالة التي ينبغي الوصول إليها إلا من داخل الطبقة الواحدة ، أو هيكل المركبة.

```
Private element As Element
```

Dim

لأننا لا يمكن إلا إن استخدام وحدة خاصة في المستوى ، ونحن نستخدم قائمة عندما نعلن أننا بحاجة إلى عناصر داخل الأجزاء. ويمكننا أيضا استخدام قاتمة في مستوى وحده ، ومن حيث تعادل Private.

```
Dim element As Element
```


ensert.net2000@gmail.com www.Ensertnet2000.all-forum.net

الفصل الخامس

مدونة تعليقات

تعليقات المدونة هي طريقتنا في توثيق هذه المعاهدات من المدونة أن نكتب ، لمساعدتنا في فهم ما يفعل المدونة ، التي تروج لمقروءين كميات كبيرة من المدونة. إضافة إلى العديد من التعليقات ليس جيد ، رغم ذلك ، ويمكن للفوضى الشفرة ، لذلك نحن فقط استخدام التعليقات حيث سيكون عملي.
وفي Visual Basic التي نستخدمها 'رمزا لكتابة التعليقات المدونة ، على سبيل المثال :

```
'Set my text to a different value.
Me.Text = "This Is My Text"
'Close this window.
Me.Close()
```

ويمكننا أيضا إضافة تعليقات في نهاية الخط ، والتي يمكن أن تؤدى إلى إنقاذ الفضاء.

```
Me.Text = "This Is My Text" 'Set my text to a different value.
Me.Close()'Close this window.
```

عندما أنت الكتابة مشروع كبير ، وغيرها من الشعب التي قد تضطر إلى العمل على أو الحفاظ عليه ، ومن الضروري أن تقوم إضافة التعليقات ، حتى يتسنى لهم فهم ما كنت أفكر في ذلك الوقت ، ولما كنت تحاول أن تفعل مع القطع الخاصة بك مدونة.

وهناك طريقة أخرى لجعل Code أكثر قراءتها هو أنها مجموعة إلى المناطق. إذا كان لنا أن نلقي نظرة على بيئة تطوير متكاملة ، وسوف نرى أن الطبقات وغيرها من التركيبات يمكن مطوية.

```
⊟Class DoStuff

 End Class
```

ال - علامة يمكن النقر فوقه ، مما يخلق أكثر من نسخة الميثاق الطبقي.

```
⊞Class DoStuff ...
```

ولكننا لا نستطيع المضي خطوة ابعد ، وأرفق هذا بناء المنطقة باستخدام
Region…#End Region#

```
#Region "Class that does stuff"
Class DoStuff

End Class
#End Region
```

ونحن الآن عندما إضعاف عنه ، حصلنا على أكثر عرضا وصفيا.

```
Class that does stuff
```

وهذا يساهم في خلق مزيد من المدونة.

الفصل السادس

السيطرة على تدفق

ما هو "السيطرة على تدفق"؟

مراقبة تدفق سيلة لتنظيم إعدام بنات المدونة في التطبيقات الخاصة بك. دون مراقبة تدفق طلباتنا سوف تدار من السطر الأول من المدونة حتى النهاية ، تفعل نفس الشيء بالضبط في كل مرة ، ونحن لن تكون قادرة على تغيير الطريقة التي تدير البرنامج خلال وقت التشغيل. مع مراقبة البيانات يمكنك أن تكتب التطبيقات التي تجعل القرارات ، واكرر الإجراءات حسب الظروف ، التي هي سمة أساسية من سمات وجوه جميع لغات البرمجة الموجهة.

GOTO

فروع هذا البيان إلى خط المدونة داخل الطريقة الحالية. ومن استخدمها أول خلق علامة في مكان ما من الطريقة الحالية ، تليها القولون ، وبعد ذلك تنفيذ "Goto labelname". العلامة قد يكون قبل أو بعد "labelname Goto" ، ولكن يجب أن تكون داخل الطريقة الحالية.

```
GoTo Label1
'Code placed here will not execute.
Label1:
'Code placed here will execute.
```

أوامر التكرار

If...Then...Else

هذا البيان بشكل مشروط ينفذ Code تبعا لقيمة A التعبير.

هناك طرق عديدة لاستخدام هذا البيان.

```
If condition Then
    'some code here
ElseIf condition Then
    'some code here
Else
    'some code here
End If
```

```
If condition Then
    'some code here
Else
    'some code here
End If
```

```
If condition Then
    'some code here
ElseIf condition Then
    'some code here
End If
```

```
If condition Then
    'some code here
End If
```

```
If condition Then 'some code here Else 'some code here
```

```
If condition Then 'some code here
```

على سبيل المثال ، أن نقول أننا نحتاج إلى تنفيذ Code ، وهذا يتوقف على قيمة متغير اسم stringvalue. إذا كانت قيمة stringvalue هو أن أيا من القيم المتوقعة ، ثم فرع آخر من ينفذ الكود.

```
If stringValue = "STRING1" Then
    'code group 1
ElseIf stringValue = "STRING2" Then
    'code group 2
Else
    'code group 3
End If
```

في هذا المثال ، إذا كانت قيمة stringvalue يساوي "string1" ثم "مدونة المجموعة ١" ستنفذ وإذا كانت القيمة هي "string2" ثم "code group ٢' " ستنفذ ، وخلاف ذلك 'code group 3 ستنفذ .

وإذا كنا للتو اللازمة لتنفيذ مدونة تبعا إذا stringvalue يساوي "string1" ثم أننا سوف تستعين التالية التنفيذ.

```
If stringValue = "STRING1" Then 'some code here
```

إذا نحن بحاجة أيضا لتنفيذ مدونة إذا لم تكن القيمة المتساوية إلى "string1" ثم أننا يمكن استخدام التالي التنفيذ.

```
If stringValue = "STRING1" Then 'some code here Else 'some code here
```

عليك الآن أن نرى كيف يمكن ربما من المفيد لو...... ثم أمر آخر ، وتعلمت كل الطرق التي يمكنك تنفيذها. ليس هناك أي سبيل للإفلات من البيانات مشروطة ؛ أنكم على الأرجح استخدامها في كل تطبيق أو قطعة من المدونة تكتبها.

Select Case

ينفذ هذا البيان واحدة من مجموعات كثيرة من البيانات ، وهذا يتوقف على قيمة تعبيرا. عندما أنت فحص واحد متغير بالنسبة للعديد من قيم مختلفة ، ومن أكثر كفاءة لاستخدام هذه البيانات.

```
Select Case expression
   Case expression1
       'some code here
   Case expression2
       'some code here
   Case Else
       'some code here
End Select
```

بعض الامثله المدونه باستخدام الخيوط…

```
Select Case stringValue
    Case "All"
        'some code here
    Case "Your"
        'some code here
    Case "Base"
        'some code here
    Case Else
        'some code here
End Select
```

بعض الامثله على الأعداد الصحيحة باستخدام المدونة…

```
Select Case integerValue
    Case 0
        'some code here
    Case 1
        'some code here
    Case 2
        'some code here
    Case 3
        'some code here
    Case Else
        'some code here
End Select
```

يمكننا استخدام أكثر من قيمة واحدة في حالة الكتلة جدا…

```
Select Case integerValue
    Case 0, 1, 2
        'some code here
    Case 3
        'some code here
    Case Else
        'some code here
End Select
```

أشعار أول كتلة "Case ٠ ، ١ ، ٢" الشيكات لكل واحد من هؤلاء الثلاثة الممكنة القيم (٠ ، ١ ، ٢) ، والتي يمكن إن تكون مفيدة للغاية في بعض الحالات. الاكواد ذاته ، كما يمكن أن تكون مكتوبة باستخدام "إلي".

```
Select Case integerValue
    Case 0 To 2
        'some code here
    Case 3
        'some code here
    Case Else
        'some code here
End Select
```

إذا لم تكن أي من القيم في حالة القطع متساوون إلى التعبير ، ثم المدونة داخل "Case Else" ستنفذ ، وإذا كان خلاف ذلك "Case Else" لا وجود له ، ثم لا يزال كود الإعدام فورا بعد "End Select".

ويمكننا أيضا استخدام هو المقارنة مع مشغل (انظر : "مقارنة بين المشغلين").

```
Select Case integerValue
    Case Is > 1
        'some code here
    Case Is < 1
        'some code here
End Select
```

أول حالة كتلة ينفذ إذا integervalue اكبر من ١ ، والعكس بالعكس.

الحلقات التكرارية

For...Next

ويكرر هذا البيان على قسم من المدونة عددا معينا من المرات.

```
For counter As datatype = start To end Step stepValue
    'some code here
    Exit For
    'some code here
Next counter
```

ويجب أن تكون قيمة عدديه ، الذي هو المتغير الذي سيستخدم لمدونة في الحلقة. الحلقة ستستمر حتى قيمة "counter" ، وهي تساوي قيمة "end".

"Step stepValue" هو اختياري (defaults to -1)١ ، وإنما هي أيضا مفيدة للغاية. كيف تعمل الدوارة هو المتغير "counter" تبدأ على القيمة التي حددها "start" ويعمل في طريقها من خلال هذا الكود إلى أن تصل إلى "Next" ، ثم يضيف قيمة

"stepvalue" إلى "counter" ويستمر من بداية مرة أخرى.
أقول لدينا مجموعة من الخيوط يسمى "stringvalues ()" ، مع عشرة الخيوط المختلفة ، ويتعين علينا تحويل كل الخيط إلى حرف كبير.

```
For arrayIndex As Integer = 0 To 9
    ConverToUpperCase(stringValues(arrayIndex))
Next counter
```

في هذا المثال نترك اصل "Step stepValue" أننا باستخدام الافتراضي stepvalue ، الذي هو دائما "+١".

ماذا سيحدث هنا هو الكود يخوض لمرة واحدة كل قيمة من ٠ إلى ٩ (وهو ما قد يعني - في المدونة بين ستنفذ ١٠ مرات) ، incrementing arrayindex في كل مرة ، ونحن نستخدم arrayindex إلى مرجعيه البند في طائفة stringvalues مع فهرس arrayindex.

إذا أردنا الذهاب إلى الوراء من خلال stringvalues (من العاشرة البند إلى الأولى البند (ثم أننا سوف تعدل الدوارة لجعله يذهب من ٩ إلى ٠ ، وإعطائها stepvalue من -١.

```
For arrayIndex As Integer = 9 To 0 Step - 1
    ConverToUpperCase(stringValues(arrayIndex))
Next counter
```

كما أننا قد نحتاج إلى بذل كل ثانية فقط قيمة حرف كبير.

```
For arrayIndex As Integer = 0 To 9 Step + 2
    ConverToUpperCase(stringValues(arrayIndex))
Next counter
```

وكما ترون ، وهذه الأنواع من الحلقات مفيدة للغاية.

Do...Loop

ويكرر هذا البيان فرعا للكود ، في حين أن شرط منطقي صحيح أو حتى يصبح الشرط صحيحا.

وهناك طريقتان لاستخدام هذا البيان ، ولك أن تختار الذي يلائم لكم أفضل.

```
Do While or Until Condition
    'some code here
    Exit Do
    'some code here
Loop
```

```
Do
    'some code here
    Exit Do
    'some code here
Loop While or Until Condition
```

الشرط يجب تقييم منطقي إلى التعبير.

اعتمادا علي إذا كنت تستخدم أو حتى حين ، في أي شيء - بين "Do" Loop "سوف تتكرر حتى إما الشرط إلى نتائج حقيقية أو كاذبة ، واحد فقط من حين إلى حين ، أو يمكن استخدامها.

بينما إذا استخدمت ، ثم الحلقة سيكرر حتى قيمة الشرط ، وهي تساوي كاذبة.
حتى إذا استخدمت ، ثم الحلقة سيكرر حتى قيمة الشرط هو الصحيح.

"Exit Do" هو اختياري ، ونقل السيطرة على الخروج من الحلقة ، واستمرار تنفيذ المدونة مباشرة بعد الحلقة.

For Each...Next

ويكرر هذا البيان على قسم من المدونة عن كل عنصر في مجموعة.

```
For Each element As dataType In group
    'some code here
    Exit For
    'some code here
Next element
```

"As dataType" هو اختياري.
"Exit For" هو اختياري ونقل السيطرة للخروج من الحلقة ، واستمرار التنفيذ على الفور بعد Next.

تخيل لدينا مجموعة من الخيوط ، stringvalues () ، الذي يتضمن عشر الخيوط ، ونحن بحاجة إلى تحويل كل الخيط إلى الاعلى القضية.

ونحن سوف تتخذ طائفة ، ولها تعديله في كل Each…Next مثل هذا…

```
For Each stringValue As String In stringValues
.convert the value of stringValue to upper case text'
Next stringValue
```

أو أننا يمكن أيضا أن تفعل نفس الشيء مثل هذا…

```
Dim stringValue As String
For Each stringValue In stringValues
    'convert the value of stringValue to upper case text.
Next stringValue
```

وكما لاحظت لديك على الأرجح ، والطريقة الأولى هي إحدى اقل من خط المدونة ، والتي اعتقد شخصيا هو أكثر بكثير مما يمكن قراءتها الثانية ، ولكن هذا أمر متروك لكم. الثاني متغير يحتاج المرء إلى أن يكون خلق مسبقا (stringvalue).

While...End While

هذا الهيكل ينفذ سلسلة من المدونة ما دام الشرط صحيح.

```
While condition
    'some code here
    Exit While
End While
```

الشرط يجب تقييم لقيمة منطقي. "Exit While" هو اختياري ، ومخارج هيكل ، واستمرار كود التنفيذ فورا بعد "End While".

With...End With

هذا البيان الذي ينفذ سلسلة من التصريحات المتكررة جعل الإشارة إلى واحد من وجوه.

دعنا نقول لدينا جسم الثلاجة التي دعت بعض الممتلكات. الوصول إلى كل الممتلكات كنا لنوع kitchen.property right ؟

```
Fridge.AppleCount = 1
Fridge.BananaCount = 2
Fridge.OnionCount = 5
Fridge.MilkLitres = 6
Fridge.EggCount = 12
```

خطأ. يمكننا الآن استخدام With…End مع ذلك أن البيانات ليست لدينا كود لقدر.

```
With Fridge
    .AppleCount = 1
    .BananaCount = 2
    .OnionCount = 5
    .MilkLitres = 6
    .EggCount = 12
End With
```

أساسا يأخذ بعيدا الحاجة إلى نوع "Fridge " وبدلا من ذلك انك لا تملك إلا ليصل إلى "."

وكما ترون ، وبهذه الطريقة هو أسهل بكثير وأسرع. لا تنسوا إن هذه التصريحات ؛ أنها ستساعدك بها كثيرا.

عشعشه هياكل المراقبة

يمكنك وضع هياكل المراقبة داخل هياكل المراقبة الأخرى. A مراقبة وضعت داخل بيان آخر هو السيطرة على البيان قال إن "nested ".

في هياكل المراقبة Visual Basic يمكن متداخل إلى ما يصل إلى المستويات التي ترغب فيها ، والتي هي مفيدة جدا ولكن يمكن أيضا أن تخفض المقروءين.

```
If 1 = 2 Then
    If 2 = 3 Then
        If 3 = 4 Then
            If 4 = 5 Then
                'This will never happen!?
            End If
        End If
    End If
End If
```

```
If 1 = 2 Then
    Select Case 1
        Case 2
            Select Case 2
                Case 1
                Case 3
            End Select
        Case 3
    End Select
End If
```

العشعشه ستثبت أنها مفيدة جدا في كثير من الظروف المختلفة.

الخروج مراقبة البيانات

يمكنك استخدام "Exit " البيان إلى الخروج من اصل معظم هياكل المراقبة (Do loops, For loops, Functions, Properties, Select statements, Subs, Try Catch blocks or While loops)

الخروج (Do or For or Function or Property or Select or Sub or Try or While)

على سبيل المثال ، اسمحوا خروج من اصل sub…

```
Sub DoStuff()
    'Code here will be executed.
    Exit Sub
    'Code here will not be executed.
End Sub
```


ensert.net2000@gmail.com

الفصل السابع

الأساليب

ما هي "الأساليب"؟Methods

لجعل البرمجة أكثر نموذجية ، يمكن قراءتها وإعادة استخدامها ، لدينا شيء يسمى الأساليب التي في حوزتنا.

دعنا الآن أن تنظر آلة تصوير مستندات هو طريقة. ولدينا شركة كبيرة ، مع العديد من المستخدمين أن الحاجة إلى جعل جميع نسخ. أليس من المنطقي أكثر أن يكون سوى إله واحد وجميع العاملين في استخدامها ، من اجل أن يكون لها ناسخة لكل موظف؟

تخيل لدينا أي طلب يمكن أن تفتح الملفات والاستيراد (منفصلتين قائمة الخيارات). ويمكن أن يكون لنا طريقة واحدة ، ربما اسمه showopenfiledialog ، الذي يظهر مربع حوار مستعرض الملف تطلب من المستخدم لاختيار ملف ، بدلا من الاضطرار نفس الكود في اثنين من أماكن مختلفة.

```
Sub ShowOpenFileDialog()
    'Show an open file dialog
End Sub
```

المهام والوظائف الفرعية

وفي Visual Basic لدينا نوعين من الأساليب ، ومهام الوظائف الفرعية. والفرق الرئيسي بينهما هو أن احد عوائد بقيمة (Function) ، والآخر لا (Subroutine) ، ولكن كل منهم ما زال الأسلوب.

في Visual Basic نعلن روتين ثانوي مثل هذا :

```
Sub SubRoutineName()

End Sub
```

نحن تضاف مدونة للروتين ثانوي في الفترات الفاصلة بين "Sub SubRoutineName " و " End Sub ".

في Visual Basic نعلن دالة مثل هذا :

```
Function FunctionName() As DataType

End Function
```

وكأنه روتين ثانوي ، ونحن تضاف مدونة لهذه الوظيفة في الفترات الفاصلة بين " Function FunctionName " و "End Function ".

DataType هو نوع من البيانات إن مهمة العودة. على سبيل المثال القادم وظيفة العودة قيمتها مع البيانات نوع الخيط.

```
Function FunctionName() As String

End Function
```

على سبيل المثال ، قد نكون في يوم من الأيام أن يطلب من مدونة طريقة أن يسترجع اسم المستخدم من الشخص الذي يستخدم هذا الكمبيوتر. ونحن من شأنه أن يخلق دالة على أن عودة سلسلة القيمة التي تحتوي على اسم المستخدم.

في ما يلي المدونة نخلق سلسلة القيمة ودعا اسم المستخدم ، ثم وضعنا قيمة له. ونحن نستخدم "Return " إلى تل Visual Basic مجموعة قيمة من وظيفة إلى القيمة المحددة ، وهي في هذه الحالة هو "اسم المستخدم".

```
Function GetUserName() As String
    Dim userName As String
    'Insert code that retrieves the username of
    'the current user, and sets “userName” to that value.
    Return userName
End Function
```

الشفرة التالية عندئذ خلق القيمة ودعا اسم المستخدم ، وبعد ذلك الدعوة إلى وظيفة getusername لاسترجاع تلك القيمة.

```
Dim userName As String = GetUserName()
```

لكود الإجراءات الروتينية التي لا تحتاج إلى أن تكون قيمة عاد نحن نستخدم الوظائف الفرعية. على سبيل المثال قطعة من المدونة أن يغلق أي طلب يمكن أن يكون روتين ثانوي دعا shutdownapp.

```
Sub ShutDownApp()
    'Insert code to close any open files.
    'Insert code to close the main window.
End Sub
```

إشعار بأن روتين ثانوي لا "Return " أي شيء.

طريقة بارامترات

ويمكننا أيضا تمرير البارامترات إلى المهام والوظائف الفرعية. أضفنا معالم في الفترات الفاصلة بين الأقواس للتو بعد طريقه ، ومع فواصل تفصل بينهما.

```
Function DoStuff(ByVal Param As String) As String

End Function
```

```
Sub DoStuff(ByVal Param1 As String, ByVal Param2 As String)

End Sub
```

Byval مقابل byref

Byval عادة يمر نسخة من الجسم ، في حين byref يمر فعلية أو وصلة الإشارة إلى أن وجوه.

النظر في المثال التالي…

هذه الطريقة لها بارامتر اسم النص الذي اقر باستخدام byval.

```
Sub ModifyText(ByVal Text As String)
    Text = "THE MODIFIED TEXT"
End Sub
```

وهذه الطريقة ايضا لها بارامتر اسم النص ، بل إنها مرت باستخدام byref بدلا من ذلك.

```
Sub ModifyText(ByRef Text As String)
    Text = "THE MODIFIED TEXT"
End Sub
```

عندما ندعو الشفرة التالية باستخدام أول byval طريقه ، "Text " لن يتم تعديلها ، ولكن عندما ندعو مستخدما الثانية byref طريقه ، "Text " وسوف يعدل.

```
Dim Text As String = "THE UNMODIFIED TEXT"
Me.ModifyText(Text)
Text = ?
```

ذلك أساسا إذا كنت في حاجة إلى تعديل قيمة بواسطة طريقة ، ثم لك أن تمر قيمة باستخدام byref ، وألا تمر عليك مستخدما byval (وهو الخيار الافتراضي).

إثقال أساليب

في .Net ومن الممكن الآن "Overload " الأساليب ، التي يمكن أن تكون قوية جدا إذا كنت تستخدم أنها حق. كالزائد هو أساسا وسيلة لخلق اثنين (أو أكثر) من الطرق ، الذي يحمل نفس الاسم ، ولكن لها بارامترات مختلفة ، وشفره مختلفة.

دعنا نقول ، على سبيل المثال ، أن لدينا والتي يمكن تطبيقها صفحة من النصوص المطبوعة ، ونحن نريد من المستخدم أن تكون قادرة على طباعة صفحة واحدة ، أو كل الصفحات. نحن سيخلق الرئيسية روتين ثانوي فقط أن تطبع كل الصفحات ، ومثقلا النسخة التي تطبع فقط الصفحة التي يحددها pagenumber. إنها سوف تبحث في هذا الشكل.

الرئيسية روتين ثانوي...

```
Sub Print()
    'Print all pages.
End Sub
```

فان مثقلا روتين ثانوي...

```
Sub Print(ByVal PageNumber As Integer)
    'Print only the page specified in PageNumber
End Sub
```

الآن عندما يحين وقت المطبوعة ، ونتحقق لذا كان المستخدم مختارة الصفحة ١ ، على سبيل المثال ، ثم ندعو Print(1) لطباعته ، أو إذا لم يحدد المستخدم صفحة مطبوعة ، ونحن ندعو Print(1) مع آي بارامترات وجميع صفحات مطبوعة.

كما يمكنك أن ترى على الأرجح ، هو إثقال أساليب عظيم طريقة لإيجاد كود أكثر مرونة.

الفصل الثامن

الطبقات

ما هو "Class" ؟

الطبقة هو أفضل وصف بأنه مخطط للجسم الذي سيتم إنشاءه إثناء التنفيذ. فهو يحدد الممتلكات والطرق التي يمكن أن يستخدمها الجسم.
"Instantiation " هو أننا عندما خلق حالة من درجة في الذاكرة.

ونحن في تحديد الطبقة Visual Basic مثل هذا :

```
Class ClassName

End Class
```

مدونة لدينا ونحن في غنى عن الطبقة - بين "Class ClassName " و "End Class " (يستعاض classname مع اسما وصفيا من اختيارك).

دعنا نحاول التفكير في سيارة بوصفها الطبقة.

```
Class Car

End Class
```

وسنعمل على تهيئة بعض المتغيرات داخل هذه الطبقة الجديدة لتخزين اسم اللون والدولة من السيارة.

```
Class Car
    Public Name As String = "Toyota"
    Public Colour As Color = Color.Green
    Public IsCarStarted As Boolean = False
End Class
```

ألان يمكننا أن نخلق بعض الأساليب لبدء وإيقاف السيارة ، التي حددت قيمة iscarstarted إلى صحيحا أو غير صحيح.

```
Class Car
    Public Name As String = "Toyota"
    Public Colour As Color = Color.Green
    Public IsCarStarted As Boolean = False

    Public Sub StartCar()
        Me.IsCarStarted = True
    End Sub

    Public Sub StopCar()
        Me.IsCarStarted = False
    End Sub
End Class
```

لاستخدام هذا كائن جديد في المدونة أعمالنا ، ونحن instantiate عليه ، ثم تحدد ممتلكاتها و البدء فيه .

```
Dim myCar As New Car
myCar.Name = "Toyota Conquest"
myCar.Colour = Color.White
myCar.StartCar()
myCar.StopCar()
```

الطبقة البناءون

الطبقات كما بنينا في أساليب يسمى "Constructors " التي تنفذ على خلق من الطبقة . وفي Visual Basic التي نستخدمها الفرعي New() لتحديد البناء .

```
Class Car
    Sub New()
    End Sub
End Class
```

فرعية جديدة يمكن ايضا أن تكون محملة بأكثر من طاقتها ، مما يتيح لك جعل البناء أسهل بكثير ، على سبيل المثال : A أضفنا البناء مع المعايير التي ستحدد المعلومات عن السيارة بالنسبة لنا .

```
Class Car
    Public Name As String
    Public Colour As Color
    Public IsCarStarted As Boolean = False

    Sub New(ByVal Name As String, ByVal Colour As Color)
        Me.Name = Name
        Me.Colour = Colour
    End Sub

    Public Sub StartCar()
        Me.IsCarStarted = True
    End Sub

    Public Sub StopCar()
        Me.IsCarStarted = False
    End Sub
End Class
```

الآن ليس لدينا إلى المحددة يدويا خواص السيارة وجوه ، ونحن مجرد إضافتها كما في معالم البناء.

```
Dim myCar As New Car("Toyota Conquest", Color.White)
myCar.StartCar()
myCar.StopCar()
```

الطبقة الميراث

ويمكن ايضا أن ترث الطبقات من الطبقات الأخرى ، وهي مفيدة للغاية في كثير من الظروف.

دعونا نقول إننا نريد لتخزين معلومات عن الدراجات النارية الآن. انه سيكون في حاجة لتخزين المعلومات على النحو نفسه هل السيارة وأيضا مبلغ عجلات الدراجة له.
ونحن أول خلق طبقة تسمى دراجة نارية ، وراثة من السيارة وتضيف الطبقة متغير لمبلغ من العجلات.

```
Class MotorBike
    Inherits Car
    Public Wheels As Integer
End Class
```

ثم علينا أنشاء لجنة جديدة البناء ، مع أضافيا بارامتر للعجلات.

```
Class MotorBike
    Inherits Car
    Public Wheels As Integer
    Sub New(ByVal Name As String, ByVal Colour As Color, ByVal Wheels As
Integer)
    End Sub
End Class
```

وأخيرا ، نطالب الأصلي للسيارة من الدرجة البناء بكتابة mybase.new () مع مجموعة المعالم وجهة نظرنا الجديدة المتغيرة لقيمة العجلات. Mybase يتيح لك الوصول إلى أساليب وممتلكاتهم مباشرة من الدرجة بأن ما تتمتعون به من الدرجة يرث ، وهو في هذه الحالة أمر السيارة الطبقة.

```
Class MotorBike
    Inherits Car
    Public Wheels As Integer
    Sub New(ByVal Name As String, ByVal Colour As Color, ByVal Wheels As
Integer)
        MyBase.New(Name, Colour)
        Me.Wheels = Wheels
    End Sub
End Class
```

وكما ترون ، عندما كنت وراثة من رتبة أخرى ، انك نسخة جميع الطرق والأملاك منه.

الفصل التاسع

الخصائص Properties

ما هي "Properties"؟

الخصائص هي أساسا نفس المتغيرات ألا انه يمكننا أن تعديل كود يحدث عندما حصلنا على قيم أو مجموعة منهم. أنها يمكن أن تكون مفيدة جدا في حفنة من الظروف.

باستخدام الخصائص

لو كان لدينا متغير دعا currenthour ، التي تخزن ساعة الراهنة ، وإننا وردا على سؤال حول هذا المتغير بعد ساعة ولقد خلقنا ومن ثم فانه لن يكون من حق لم يعد.

وفي حالات مثل هذه الخصائص يمكن أن تكون مفيدة جدا.

ولندع إيجاد الخصائص للcurrenthour :

```
Property CurrentHour() As Integer
    Get

    End Get
    Set(ByVal value As Integer)

    End Set
End Property
```

إننا لن نحتاج إلى مجموعة قيمة من currenthour ، حتى يمكننا أن نجعل الخاصية للقراءة فقط ، وذلك باضافه للقراءة فقط وإزالة حدد طريقة :

```
ReadOnly Property CurrentHour() As Integer
    Get

    End Get
End Property
```

الآن لدينا جميعا أن نفعله هو أن احصل على ساعة الحالية والعودة إليها.

```
ReadOnly Property CurrentHour() As Integer
    Get
        'Create a variable to store the hour.
        Dim hour As Integer
        'Set hour to the current hour.
        hour = Today.Hour
        'Return the value.
        Return hour
    End Get
End Property
```

وألان أصبحت لدينا متغير أن يمثل السليم الحالية ساعة.

دعنا نحاول انه مع شيء آخر : ربما كان لدينا نافذة مع أنها قائمة على ونريد للمستخدم لتكون قادرة على إخفاء أو إظهار القائمة.

أولا نحن إنشاء منطقي الممتلكات دعا شيئا كهذا menuvisible ، ومتغير لتخزين أكثر مما هو منطقي القيمة الحقيقية لمجموعة لأن القائمة هي مرئية بشكل افتراضي.

```
Dim isMenuVisible As Boolean = True
Property MenuVisible() As Boolean
    Get

    End Get
    Set(ByVal value As Boolean)

    End Set
End Property
```

أول مدونة لأضفنا أن أحصل على العودة ismenuvisible. الثانية أضفنا مدونة لمجموعة قيمة الشيكات التي مرت عليه ، وتبين القائمة أو الجلود ومجموعات قيمة ismenuvisible تبعا لذلك.

```
Dim isMenuVisible As Boolean = True
Property MenuVisible() As Boolean
    Get
        Return isMenuVisible
    End Get
    Set(ByVal value As Boolean)
        If value = True Then
            'Show the menu.
            'Set the value of isMenuVisible
            isMenuVisible = True
        Else
            'Hide the menu.
            'Set the value of isMenuVisible
            isMenuVisible = False
        End If
    End Set
End Property
```

الفصل العاشر

المصفوفات

ما هي "المصفوفات" Arrays؟

جميع لغات البرمجة لها تنفيذ بعض الصفوف.
طائفة هي في الأساس قائمة من البنود. واضح من استخدام الصفوف عندما يكون لديك لتخزين الأشياء اثنين أو أكثر من نفس النوع. فعلى سبيل المثال لو كان لديك واحد لتخزين آلاف شريطا القيم ، فإنه سيكون بمثابة كابوس لإنشاء لجنة جديدة للمتغير كل بت من البيانات!

```
Dim stringValue1 As String
Dim stringValue2 As String
Dim stringValue3 As String
'Imagine typing this 1000 times!
```

لهذه الحالات نقوم بإنشاء طائفة - قائمة الأشياء المخزنة تحت اسم واحد ، ويمكن الوصول أليها مع كل مؤشر.
في Visual Basic صفوف دائما أن نبدأ من الصفر (١٠ بنود في طائفة = ٠ إلى ٩).
فنحن نصنع طائفة مثل هذا…

```
Dim array1(UpperBound) As dataType
```

أو ما شابه ذلك…

```
Dim array1 As dataType(UpperBound)
```

أو يمكننا أن يسند البنود إلى طائفة الحق في بداية مثل هذا…

```
Dim array1() As dataType = New dataType() {object1, object2,object3}
```

أو ما شابه ذلك…

```
Dim array1 As dataType() = New dataType() {object1, object2, object3}
```

محصور في أعلى هو مؤشر آخر بند في مصفوفة ، فلنقل نقوم بإنشاء طائفة مع عشرة عناصر ، محصور في أعلى ستكون تسعة لأننا دائما إن نبدأ من الصفر. محصور في أعلى كما هو اختياري ، ويمكننا تغيير طائفة وفي وقت لاحق في المدونة ، إذا لزم الآمر.

أنا الآن في المشي لكم من خلال أساسيات الصفوف.

ولإيجاد مجموعة جديدة ، ويسميها stringvalues. سنكون تخزين خمسة قيود في أنها ، لذلك نحن مجموعة upperbounds إلى ٤.

```
Dim stringValues(4) As String
```

الآن سنقوم إسناد القيم إلى كل بند في مصفوفة.

```
stringvalues(0) = "String Value 0"
stringvalues(1) = "String Value 1"
stringvalues(2) = "String Value 2"
stringvalues(3) = "String Value 3"
stringvalues(4) = "String Value 4"
```

الآن كل بند في مصفوفة لديها قناعتها القيمة.

ونحن نقول الآن يريدون إضافة بند آخر إلى طائفة ، أو تغيير حجم طائفة. هذا أمر سهل إلى حد ما في Visual Basic. ونحن نستخدم الكلمات الرئيسية redim لتغيير حجم مجموعة من مثل هذا...

```
ReDim stringValues(5)
```

الآن لدينا مجموعة متنوعة يمكن أن تعقد ستة بنود ، ولكن المشكلة مع redim هو أن ككل هو خلق طائفة جديدة ، بحيث أنها لا تحتوي على أي من القيم في مصفوفة السابقة.
لا مشكلة. يمكننا أن نستخدم الكلمات الرئيسية في المحافظة جنبا إلى جنب مع redim ، التي فعلا يخلق مجموعة جديدة وثم نسخ بنود من النظام القديم إلى الجديد طائفة طائفة.

```
ReDim Preserve stringValues(5)
```

إذا كنت تفعل ولكن تغيير حجم طائفة مع redim الحفاظ على صغر حجمها ، ثم أي من البنود التي لا تندرج في سيتم chopped.

ملاحظه : redim المحافظة يمكن أن تكون مروعه فقدان الأداء عند العمل مع صفوف كبيرة ، كما هو الجامع طائفة نسخها إلى مجموعة جديدة في كل مرة لكم استخدامها. حيثما أمكن ، في محاولة لعدم استخدام redim المحافظة.

في أي حال ، لا إلى الوراء مثلا. يمكننا الآن أن يضاف بند جديد إلى طائفة

```
stringvalues(5) = "String Value 5"
```

سأقوم الآن الخوض في بعض الوظائف المفيدة التي هي في صلب طائفة وجوه.

بعض مفيدة المدمج في طائفة وظائف

Array.copy

نسخ مجموعة من العناصر من مجموعة متنوعة بدءا من العنصر الأول والمعاجين منهم طائفة إلى أخرى ابتداء من أول عنصر.

```
Array.Copy (sourceArray, sourceIndex, destinationArray,
destinationIndex, length)
```

نسخ مجموعة من العناصر بدءا من طائفة محددة المصدر في مؤشر والمعاجين منهم طائفة إلى أخرى ابتداء من الوجهة المحددة في الرقم القياسي.

Array.CopyTo

```
Array.CopyTo (array, index)
```

نسخ جميع عناصر الحالية ذات بعد واحد إلى طائفة محددة ذات بعد واحد ابتداء من طائفة على طائفة مؤشر الوجهة المحددة.

Array.indexof

```
Array.IndexOf (array, value) As Integer
```

عند البحث عن وجوه والمحدد ليعيد بذلك الرقم القياسي من وقوع الأولى ضمن كامل الأبعاد طائفة واحدة. وإذا كان الجسم المحدد لم يتم العثور بعد ذلك indexof عائدات بقيمة -١.

```
Array.IndexOf (array, value, startIndex, count) As Integer
```

عند البحث عن وجوه والمحدد ليعيد بذلك الرقم القياسي من وقوع الأولى ضمن مجموعة من العناصر في مصفوفة البعد الواحد التي تبدأ إعمالها في مؤشر محدد وتتضمن عددا محددا من العناصر. وإذا كان الجسم المحدد لم يتم العثور بعد ذلك indexof عائدات بقيمة -١.

Array.lastindexof

```
Array.LastIndexOf (array, value) As Integer
```

عند البحث عن وجوه والمحدد ليعيد بذلك الرقم القياسي من وقوع الماضي ضمن كامل الأبعاد طائفة واحدة.

```
Array.LastIndexOf (array, value, startIndex) As Integer
```

عند البحث عن وجوه والمحدد ليعيد بذلك الرقم القياسي من وقوع الماضي ضمن مجموعة من العناصر في مصفوفة البعد الواحد الذي يمتد من العنصر الأول إلى الرقم القياسي المحدد.

```
Array.LastIndexOf (array, value, startIndex, count) As Integer
```

عند البحث عن وجوه والمحدد ليعيد بذلك الرقم القياسي من وقوع الماضي ضمن مجموعة من العناصر في مصفوفة البعد الواحد التي تحتوي على عدد محدد من العناصر والغايات المحددة في المؤشر.

Array.length

يحصل ٣٢ بت عدد صحيح أنه يمثل العدد الكلي للعناصر في كل الأبعاد للطائفة.

Array.reverse

```
Array.Reverse (array)
```

الانتكاسات تسلسل العناصر في بأسره بعد واحد طائفة.

```
Array.Reverse (array, index, length)
```

عكس سلسلة من العناصر في مجموعة من العناصر في مصفوفة أحاديه البعد.

Array.sort

```
Array.Sort (array)
```

أنواع العناصر في كامل الأبعاد طائفة واحدة.

الفصل الحادي عشر

الأحداث

ما هي "الإحداث"؟

الأحداث يمكن أن يكون أفضل وصف بأنه رد فعل على ما حدث للتو.
تصوروا أننا قد الهواتف بدون قارع الأجراس ، وكانت لدينا باستمرار لالتقاط جهاز الاستقبال والتأكد من أن احدهم يتحدث عن الجانب الآخر. رنين الهاتف هو حدث أن يخطر لنا أن ثمة داعيا.
يمكننا أن تطبق على وجه الدقة المنطق نفسه على الأحداث في البرمجة.
لو أننا نبذل البرنامج الذي يحتاج إلى التحقق من المدخلان لوحة المفاتيح ، على سبيل المثال ، ونحن نتعرض لفحص حالة مفاتيح على لوحة المفاتيح ، ثم برنامجنا ستتناول الكثير من معالج السلطة ، واثر بطء في أسفل كل شيء ، ولكن إذا كان برنامجنا تكمن ناءمه وانتظر إشارة يحكى أن رئيسي قد يضغط ثم أنها سوف تستعمل أي معالج السلطة على جميع!

استخدام الأحداث

حتى الآن لديكم علم السلطة من الطبقات ، وكيف يمكنك أن تصنع بنفسك الطبقات ، مع أساليب وممتلكاتهم. ونحن الآن في تعلم كيفية استخدام الأحداث في الطبقات.

ولندع النار حتى Visual basic.net.
عندما ديك البصرية الأساسية بالظهور ، انقر فوق File → New Project ، اختر "Windows" من قائمة ، والصحافة ok.
أننا ننطلق من شكل أساسي مع وجوه ، في تصميم واسطة.
على اليسار لدينا صندوق ألعده ؛ هذا يحتوي على جميع الأشياء يمكنك استخدامها في التطبيقات الخاصة بك. تفتح صندوق الأدوات من خلال عقد الماوس فوق زر ، وتجد textbox في القائمة. انقر نقرأ مزدوجا textbox ، واحدة textbox ستظهر على النموذج الخاص بك.
الآن حق انقر على النموذج الخاص بك ، واختر عرض المدونة ، أو الصحافة f7.

سترى إعلان لform1.

```
Public Class Form1

End Class
```

دعونا نأخذ لحظة لتذهب من خلال هذا الإعلان.
"Public" يقول لنا أن نتمكن من الوصول إلى هذه الدرجة من إي مكان ، و"form1" يخبرنا أن هذه الطبقة هي التي تحمل عنوان "form1".

طيب ، والآن نحن بصدد استخدام واحد من form1 للأحداث ، وحمل الحدث.
عادل المدونة أعلاه المجال ، سترون اثنين إسقاط القوائم ، إن تفتح واحدة على اليسار وانقر على (Form1 Events) ، ثم تفتح القائمة المنسدله على اليمين.

أنت سيلاحظ أن هناك ثروة من الإحداث في متناول لنا!

دعنا ننظر في بعض هذه الإحداث بسرعة.
- انقر هذه الإحداث عندما حرائق (نعم أنت مجزور عليه) ينقر مستخدم شكل مع الماوس.
- إغلاق هذا الحدث حرائق عندما ينقر مستخدم الوثيق زر على النافذة ، أو هي نافذة مغلقة في المدونة. يمكنك إيقاف شكل من أقفال هنا ، وإذا كنت بحاجة إلى ، بالإطلاع على إلغاء قيمة منطقي في canceleventargs ، الذي صدر كما هو معيار من هذا الحدث.

عاد الآن إلى المسالة المطروحة : التعليق على form1 للتحميل الحدث. اختر التحميل من إسقاط قائمة على الحق ، وإنكم سوف نلاحظ أن بعض هذه المدونة المضافة.

```
Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load

End Sub
```

هذا هو التخطيط للحمل الحدث. أشعار المقابض الكلمات الرئيسية ، والتي تأخذ في me.load بوصفها بارامتر. هذا يربط form1_load الأجراء إلى form1 للتحميل الحدث ، ودون أن يكون شيئا عاديا ولكن كل يوم طريقة.

الآن تضاف هذه المدونة داخل الحدث (بين "Private Sub " و "End Sub").

```
Me.TextBox1.Text = "Hello Events!"
```

ما ينبغي أن يحدث الآن ، هو عندما شكل الأحمال ، نص textbox1 (الذي هو textbox أضفنا في وقت سابق) وينبغي لمجموعة "مرحبا الأحداث!".
الصحافة f5 لبناء وإدارة المشروع لهذا الاختبار (أو انقر على Debug→Start Debugging).

ولندع تضاف en الحدث لtextbox1 الآن. تفتح إسقاط القائمة في أعلى اليسار ، واختر textbox1 ، وبعد ذلك تفتح القائمة على الحق واختر textchanged. هذا الحدث حرائق عندما مستخدم يعدل النص.

```
Private Sub TextBox1_TextChanged(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TextBox1.TextChanged
End Sub
```

يضاف ما يلي المدونة داخل الحدث إعلان لtextchanged.

```
Me.Text = Me.TextBox1.Text
```

هل هذا هو ما يحدد أشكال "النص" إلى قيمة الممتلكات من textbox1 نص الممتلكات. الصحافة f5 لاختبار هذا المشروع ، وأشعار ماذا يحدث عندما تكتبها النص إلى textbox1.

أنت ربما التفكير في كل الاحتمالات المتاحة لك باستخدام هذه الإحداث. هناك فعلا ما يكفي لإحداث تفعل أي شيء تقريبا. يستغرق الأمر بعض الوقت لننظر حولنا ومعرفة الإحداث هناك خيارات متوفرة لك.

خلق الإحداث

ويمكننا أيضا أن تخلق المناسبات الخاصة بنا للفئات التي ننشئها.

خلق طبقة أن تثر الأحداث

وإذا كنا قد لخلق طبقة أن يقرأ من ملف ونحن في حاجة إلى معرفة عندما انتهى كل قراءة ، ثم أننا يمكن أن يخلق حالة منطقتنا.

```
Class FileReader
    Sub StartReading
        'read data from the file
    End Sub
End Class
```

أيجاد العرف الحدث داخل الطبقة أن مثل هذا :

```
Public Event FinishedReading()
```

وبعد ذلك رفع الحدث وعندما انتهت القراءة :

```
RaiseEvent FinishedReading()
```

التزامنا الكامل الطبقة سوف تبدو هذه :

```
Class FileReader
    Public Event FinishedReading()
```

```
    Sub StartReading
        'read data from the file
        'raise the FinishedReading event.
        RaiseEvent FinishedReading()
    End Sub
End Class
```

أشعار ونحن نستخدم raiseevent لرفع حدثا داخل الطبقة الحالية.

خلق طبقة والتي تعالج الأحداث

الآن فأننا سنضع الطبقة التي تعالج الطبقة السابقة للأحداث.

خلق طبقة خاصة مع المتغير أن مخازن جديدة filereader ، والبناء أن تطلق عليه procedure filereader.startreading.

```
Class FileReaderManager
    Private myFileReader As FileReader

    Sub New()
        Me.myFileReader.StartReading()
    End Sub
End Class
```

الآن فأننا سنضع إجراءات لمعالجة finishedreading الحدث.

```
Sub FinishedReading()
    'notify the user that we have
    'read the file
End Sub
```

التزامنا الكامل الطبقة الآن يشبه هذا :

```
Class FileReaderManager
    Private myFileReader As FileReader

    Sub New()
        Me.myFileReader.StartReading()
    End Sub

    Sub FinishedReading()
        'notify the user that we have
        'read the file
    End Sub
End Class
```

ونحن الآن بحاجة إلى ربط هذا الأجراء لهذا الحدث إن myfilereader يولد. لذلك لدينا خياران :

Addhandler

ويضيف هذا المعالج للحدث ، وذلك باستخدام الأجراء أن نكون قد حددت بالفعل في منطقتنا الطبقة.

نستخدمها مثل هذا :

```
AddHandler Event, EventProcedure
```

الحدث هو الحدث ونحن على التعليق على وeventprocedure هو الأجراء ونحن على التعليق على هذا الحدث.

ذلك إننا سوف أدراجه في منطقتنا دون جديدة مثل هذا :

```
AddHandler myFileReader.FinishedReading, AddressOf Me.FinishedReading
```

حتى التزامنا الكامل الطبقة سوف تبدو هذه :

```
Class FileReaderManager
    Private myFileReader As FileReader

    Sub New()
        AddHandler myFileReader.FinishedReading, AddressOf Me.FinishedReading
        Me.myFileReader.StartReading()
    End Sub

    Sub FinishedReading()
        'notify the user that we have
        'read the file
    End Sub
End Class
```

مقابض

هذا هو أساسا نفس باستخدام addhandler ولكن في بعض الأحيان تعتبر أسهل.

القطاع الخاص myfilereader withevents كما filereader

ثم أضفنا المقابض الكلمات الرئيسية والدقيقة الحدث بأننا المناولة في نهاية اجتماعنا الأجراء :

```
Sub FinishedReading() Handles myFileReader.FinishedReading
    'notify the user that we have
    'read the file
End Sub
```

وأخيرا نحن الطبقة سوف تبدو هذه :

```
Class FileReaderManager
    Private WithEvents myFileReader As FileReader

    Sub New()
        Me.myFileReader.StartReading()
    End Sub

    Sub FinishedReading() Handles myFileReader.FinishedReading
        'notify the user that we have
        'read the file
    End Sub
End Class
```

هذا كل ما! الآن المدونة داخل هذا الأجراء سينفذ عندما filereader الطبقة يثير هذا الحدث.

الفصل الثاني عشر

الاستثناءات

هناك استثناء وجود خطأ أو سلوك غير متوقعة واجهتها إثناء تنفيذ المدونة. فعلى سبيل المثال ، إذا حاولنا قراءة من الملف الذي هو غير موجود ، ونحن الحصول على filenotfound استثناء. وإذا لم نفعل ذلك باستثناء التعامل على النحو الصحيح ، ثم طلبنا وسوف تحطم.

ونحن نستخدم منظم لمحاولة التعامل مع استثناء والصيد جميع استثناءات ممكنة ، حتى نستطيع أن يبقى برنامجنا الحالي دون التحطيم والتهيج للمستخدم.

كيف نفعل ذلك باستخدام Try...Catch...Finally البيان.

```
Try
    tryStatements
    Exit Try
Catch exception As exceptionType When expression
    catchStatements
    Exit Try
Catch
    catchStatements
    Exit Try
Finally
    finallyStatements
End Try
```

"Exit Try" هو اختياري. "باستثناء ما exceptiontype" هو اختياري.
"When expression" هو اختياري ، ويسمح الصيد كتلة تعدم إلا عندما " expression" ليقيم صحيحا.
كتل متعددة الصيد المسموح ، ودرست كل واحدة من اجل تحديد ما إذا كان يمكن معالجة النوع من الاستثناء. حتى إذا كنا مجرد استخدام الصيد دون أي نوع من الاستثناء المذكورة ، ثم أنها سوف الصيد جميع الاستثناءات.
وأخيرا ، وهو اختياري ، ودائما أعدم في نهاية الكود التنفيذ.

النظر في هذه العينات.

<u>عينه ١ :</u>

```
Try
    'Read from a file.
Catch
    'If reading from the file failed
    'then just exit.
End Try
```

عينه ٢ :

```
'Store the amount of times we have tried
'to read from a file.
Dim readAttempts As Integer = 0
tryAgain:
Try
    'Read from a file.
Catch ex As IO.FileNotFoundException When readAttempts = 0
    'The file was not on the harddrive, increment readAttempts.
    readAttempts = readAttempts + 1
    'Try again from the beginning.
    GoTo tryAgain
Catch ex As IO.FileNotFoundException When readAttempts = 1
    'We have already tried once before to read the file,
    'Show a dialog saying that we have already tried
    'once and the file probably doesn't exist.
    'Code execution will now move on to the finally block.
Finally
    'If the file was opened, then process it, otherwise
    'just exit.
End Try
```

الفصل الثالث عشر

أشكال ويندوز Windows Forms

في قلب من microsoft.net الإطار هي أشكال ويندوز ، الذي هو في جوهره بمثابة نافذة مثل النافذة التي محرر الاستخدامات. شكلا من أشكال الاعتراض هو في الواقع الطبقة ، وصمم في microsoft.net الواردة في هذا الإطار.

يمكن أن يكون شكلا من أشكال عديدة حسب الضوابط على النحو الذي تريد ، وأنها يمكن إن يكون لها مدونة خاصة بها ، والتي لكم تصميم.

ولندع فتح Visual Basic.
انقر File→New Project، اختر ويندوز والصحافة ok.
أنك ستبدأ في أشكال مصمم ، وسيكون هناك فراغ شكل إمامكم.

ننظر في بعض خواص هذا الشكل (الصحافة f4 على رأي أملاك نافذة ، أو انقر View →Properties Window).

تبدأ باختيار النص ، وتغييرها. سترون أن أشكال تعليق النص سيتغير.
القادم يتيح له الانتقال إلى backcolor الممتلكات. انقر على زر الحق (…) واختر اللون الجديد. أشكال سيتغير لون الخلفية.
محاولة تغيير التعتيم من شكل إلى ٥٠ ٪ ، انقر Debug→Start Debugging والتنقيح إشعار كيف شكل لا يمثل سوى نصف مرئية!

يمكنك أن تفعل أشياء كثيرة مختلفة مع شكل الممتلكات ، مثل تغيير خلفية الصورة (backgroundimage) ، وتجعل شكل تظهر فوق جميع أشكال أخرى (أعلى) أو تغيير أشكال الايقونه (أيقونه).

أشكال ويمكن ايضا بسهولة أنشئت في المدونة ، لأنها ليست سوى الطبقة.

```
'Create a new Form instance called testForm.
Dim testForm As New Form
'Set its text to "Test".
testForm.Text = "Test"
'Set its background colour to Green.
testForm.BackColor = Color.Green
'Display the form.
testForm.Show()
```

الـفـصـل الـرابـع عشر

system.object

مـا هـي "الأشـيـاء"؟

في Net Framework.، كل فـئـة واحـدة يـرث مـن وجوه ؛ حتى الـطـبـقـات الـتي تـصـنع نـفـسك. ومـن الـنـهـائـي في قـاعـدة الـطبـقـة. Net Framework.

تحـويـل الأجسام

آخر قـلـيـلا مـن المعـلـومـات المـفـيـدة هـو أن نـسـتـطيع تحـويـل الـطبـقـة إلى أي فـئـة عـلـى طـول طـريـقـهـا مـن المـيراث ، ويـعـود إلى حـالـتـه الاصـلـيـه الـطـبـقـة.

جـعـل مخـصـص الـطـبـقـة :

```
Class ClassA
    Public SomeNumberValue As Integer
End Class
```

خـلـق حـالـة مـن أنهـا :

```
Dim testClass As New ClassA
```

تحـويـلـهـا إلى جسم :

```
Dim objectClass As Object = CType(testClass, Object)
```

تحـويـلـهـا إلى الـوراء :

```
Dim testClass As ClassA = CType(objectClass, ClassA)
```

وجـوه أسـالـيـب

لأن جميـع الأجسام وراثـة مـن وجـوه ، كـل طـريـقـة في وجـوه الـطـبـقـة المـتـاحـة لهـم

Object.equals

تـسـتـخـدم لـلـمـقـارنـة بـين اثـنـين مـن الأجـسـام.

Object.finalize

تستخدم لتنظيف الجسم الموارد قبل وجود جسم دمرت.

Object.gethashcode

يستخدم لتوليد الرقم المميز لهذا الجسم ، بحيث يمكن استخدامه في بعثرة الجدول.

Object.tostring

تستخدم لخلق نص الخيط إن يصف الجسم.

الطاغية وجوه أساليب

وجوه أساليب يمكن بسهولة تخطيه لإيجاد الوظيفة الخاصة بنا.

ولخلق طبقة جديدة مع بعض المتغيرات.

```
Private Class TestClass
    Public Data1 As String
    Public Data2 As String
End Class
```

الآن دعونا تعلو وجوه طريقة "Equals" (يمكننا نوع "Overrides" الفضاءيه والصحافة لإظهار نافذة أكمال تلقائي ، وثم اختر "Equals").

```
Private Class TestClass
    Public Data1 As String
    Public Data2 As String
    Public Overrides Function Equals(ByVal obj As Object) As Boolean
        Return MyBase.Equals(obj)
    End Function
End Class
```

الآن نحن الطاغية "Equals" طريقة.

إزالة عودة (MyBase.Equals(obj)) (الخط) التي تقول إنها لاستخدام الافتراضي (object.equals ، وتدرج بنفسك قانون العرف ، على سبيل المثال :

```
'Convert obj to TestClass.
Dim otherObject As TestClass = CType(obj, TestClass)
'Test if I’m equal to the other object.
If (otherObject.Data1 = Me.Data1) Then
    'My Data1 is equal to the objects Data1.
    If (otherObject.Data2 = Me.Data2) Then
        'My Data2 is equal to the objects Data2.
        'Return True, because objects are equal.
        Return True
    End If
End If
'If we got here, then objects aren't equal.
Return False
```

ولتجاوز "tostring" طريقة الآن بتكرار الخطوات في المثال السابق. سنعود "data1value : data2value" كما وصف.

```
Private Class TestClass
    Public Data1 As String
    Public Data2 As String
    Public Overrides Function ToString() As String
        'Return a more descriptive string value.
        Return Me.Data1 & " : " & Me.Data2
    End Function
End Class
```

وكما ترون ، ومن السهل إلى حد ما تجاوز أساليب أعلن في وجوه الطبقة ، واستخدامها لصالحنا.

الفصل الخامس عشر

مجموعات

ما هي "مجموعات"؟

في فصل سابق علمنا عن الصفوف. صفوف تخزين البيانات ولها العديد من الوظائف المفيدة لمعالجة هذه البيانات.

الشيء الوحيد الذي يغيب عن صفوف هو انك لا تستطيع بسهولة إضافة بنود إلى طائفة ، ومن دون ريسيزينج ، الذي هو حقيقي أداء المصرف إذا كان لديك كمية كبيرة من الأشياء المخزنة في لهم .

أنني عندما بدأت أول من Visual Basic، وكنت انتظر صفوف الكثير. وكنت انتظر الكثير منهم طريقه إلى الواقع ، لأنني قد عرفوا كيف ابدأ إلى استخدام مجموعات على النحو الصحيح ، ولكن عندما فعلت أنا في النهاية معرفة كيفية استخدام هذه الجمال الترميز أصبحت اقل بكثير رتيب.

فأنا على يقين بأنني يجب أن خلقت هذا العمل البشع واحدة على الأقل ألف مرة…

```
'Need to add an item to myArray()

'Create a variable to hold the dimension
'of the new array...
Dim newArrayDimension As Integer = 0

'If myArray has items in it then use
'it's length as the new dimension, otherwise
'just use zero
If Not myArray Is Nothing Then
        newArrayDimension = myArray.Length
End If

ReDim Preserve myArray(newArrayDimension)
'Now add the item at the end.
myArray(newArrayDimension) = "New Value"
```

عندما قمت لنوي قد فعلت هذا

```
collection.Add("New Value")
```

لا ، أنا لست التنكيت.

Collections.arraylist

وهو arraylist هي أساسا مع طائفة أكثر وظيفية.

ولندع إنشاء لجنة جديدة arraylist ونرى ما يمكننا أن نفعله معها.

```
Dim list As New Collections.ArrayList
```

الآن سوف نستخدم لحلقة واحدة تضاف إليها آلاف من البنود.

```
For index As Integer = 0 To 999
    list.Add("This is String " & index.ToString)
Next index
```

طيب ، ألان لدينا قائمة تحتوي على ١٠٠٠ البنود ، واعتبر "هذا هو الخيط ٠" إلى ٩٩٩.

كيف يمكننا أن كرر خلال هذه القائمة؟

```
For Each item As String In list
    MessageBox.Show(CType(item,String))
Next item
```

تقريبا من السهل جدا! أشعار أننا نحتاج إلى استخدام ctype للتحول إلى سلسلة ، لان البند سيكون من نوع Object.

ويمكننا ايضا أن تضاف بنود معينة في مؤشر…

```
list.Insert(100, "The 100th String")
```

يمكننا بسهولة واضحة قائمة

```
list.Clear()
```

يمكننا إزالة الجسم عن طريق تمرير قيمته.

```
list.Remove("The 100th String")
```

يمكننا إزالة الجسم عن طريق تمرير مؤشرها.

```
list.RemoveAt(100)
```

يمكننا حتى إزالة مجموعة من الأشياء.

```
list.RemoveRange(100, 10)
```

إذا كان هذا لا يكفي حتى نستطيع تحويل القائمة بأكملها إلى طائفة.

```
Dim array() As Object = list.ToArray()
```

Collections.hashtable

hashtable هي قائمة من البنود التي يمكن استرجاعها عن طريق تمرير رئيسي.

المرة الأولى التي استخدم hashtable كان عندما كنت أعمل على محاولة في نسخة ويندوز اكسبلورر. لقد وجدت انه لا بأس به الأداء الخسارة إلى قراءة الأيقونات عن كل دليل والملف في كل مرة كانوا اظهروا.

ثم إنني وجدت من اصل حوالي hashtables. وأستطيع الآن تخزين الأيقونات وتمرير اسم الملف أو اسم الدليل كاداه رئيسية لاسترجاع تلك الايقونه.

ولندع إنشاء لجنة جديدة hashtable.

```
Dim hashTable As New Hashtable
```

الآن أضيف إليها بعض البنود (لكل بند من البنود يجب أن يكون مختلفا الرئيسية).

```
hashTable.Add("Icon1", Icon1)
hashTable.Add("Icon2", Icon2)
hashTable.Add("Icon3", Icon3)
hashTable.Add("Icon4", Icon4)
hashTable.Add("Icon5", Icon5)
```

الآن لاسترجاع بندا (إشعار ونحن نستخدم ctype هنا لأن hashtable.item العودة جسم لا أحد الرموز).

```
Dim Icon1 As Icon = CType(hashTable.Item("Icon1"), Icon)
```

إزالة بندا هي ايضا سهلة.

```
hashtable.Remove("File1")
```

الفصل السادس عشر
generics
مجموعات عامة

في الفصل السابق تعلمنا عن مجموعات. سأقوم ألان تبين لكم كيف أقوى المجموعات يمكن مع Visual Basic.Net سمة "generics".

عندما علمت عن مجموعات لكم ، انتم ربما لاحظ انه عندما نعقد استرجاع بندا من جمع ، ومن عاد ما جسم ، ويتعين علينا استخدام ctype لتحويلها إلى نوع محدد. هذا هو ضخم الأداء الأكل ، والطريقة الوحيدة لتصحيح هذا في النسخة السابقة من microsoft.net كان بنفسك لخلق نوع من جمع بيانات عن كل نوع لكم إن يقصد إلى متجر.

أن كان زائدا عن الحاجة. أنا أبدأ نود إعادة اختراع العجلة.

كيف generics لحل هذه؟ أنا سوف تظهر لك.

إيجاد الدرجة التي تعتزمون على تخزينها في قائمة.

```
Class MyItem
    Public Name As String
    Public Text1 As String
    Public Text2 As String
End Class
```

الآن تهيئة عامة قائمة لتخزين أعمالنا الطبقة.

```
Dim myItemList As New Generic.List(Of MyItem)
```

إنشاء لجنة جديدة للامثال بندا وإضافتها إلى القائمة.

```
Dim myItem1 As New MyItem
myItem.Name = "My Name"
myItem.Text1 = "Text 1"
myItem.Text2 = "Text 2"
```

```
myItemList.Add(myitem1)
```

استرجاع هذا البند.

```
Dim myItem1 As MyItem = myItemList.Item(0)
```

لا أكثر ctype تباطؤ أداء الشفرة وجعل لكم أكثر من نوع.

تصنيف مجموعات عامة

مفيد آخر سمة عامة المجموعات هو أن يمكنك منهم أي نوع بالطريقة التي تريدها ، باستخدام العرف مقارن.

ما نقوم به هو خلق طبقة جديدة.

```
Class MyItemListComparer
End Class
```

ثم أننا ترث من generic.comparer (Of DataType) في أعلى مرتبة.

```
Class MyItemListComparer
    Inherits Generic.Comparer(Of MyItem)
End Class
```

Visual Basic.Net ينبغي أن يضاف بعد تخطيه مقارنة الوظيفة أضفنا يرث ، والذي نقوم به عادتنا مقارنة بالدخول.

```
Public Overrides Function Compare(ByVal x As MyItem, ByVal y As MyItem) As Integer
    Return String.Compare(x.Name, y.Name)
End Function
```

الفصل السابع عشر

المندوبون

وهو مندوب هو أساسا وسيلة الربط التي مخازن ذكرى طريقة للمعالجة بحيث انه يمكن تنفيذه من مكان آخر في المدونة.

تنظر لدينا اثنين من الأجسام ، object1 وobject2.

```
Private Class Object1

End Class
```

```
Private Class Object2

End Class
```

هدفنا الرئيسي الطبقة ، والذي سنطلق mainobject ، يخلق النسخة الخاصة من كل وجوه.

```
Dim object1 As New Object1
Dim object2 As New Object2
```

الآن في مكان ما على طول الخط ، ونحن ندرك أن object1 مطلوب لاستدعاء وظيفة داخل object2. المثال التالي يبين أن هذا سيكون من المستحيل أن تفعل ، لأن لا وسيلة للربط لobject1.

```
Private Class Object1
    Public Sub DoStuff(ByVal Caller As String)
        MessageBox.Show("This procedure was called from " & Caller)
    End Sub
End Class
```

```
Private Class Object2
    Public Sub CallObject1()
        MessageBox.Show("I can't do that Dave.")
    End Sub
End Class
```

مع مندوبين ولدينا ألان وسيلة لردم هذه الهوة.

في المثال التالي أنني قد خلقت طبقة تسمى object1 ، التي لديها إجراءات دعا dostuff أن تقبل على سلسلة القيمة ودعا المكالمة. عندما dostuff يطلق عليه يظهر مربع رسالة يقول الذين أطلقوا عليه.

كما أنني قد أوجد طبقة تسمى object2 ، مع فرعي جديد أن تقبل بارامتر دعا delegatetoinvoke ، ثم تحتج فيه.

```
Private Class Object1
    Public Sub DoStuff(ByVal Caller As String)
        MessageBox.Show("This procedure was called from " & Caller)
    End Sub
End Class
Private Class Object2
    Public Delegate Sub DoStuff(ByVal Caller As String)
    Public Sub New(ByVal DelegateToInvoke As DoStuff)
        DelegateToInvoke.Invoke("Object2")
    End Sub
End Class
```

الشفرة التالية يمثل احد object1 نوع متغير اسمه "object1" ، ثم يخلق المندوب الذي هو صلة لobject1.dostuff طريقة ويمثل احد object2 نوع المتغير ، وفاة المندوب بوصفها بارامتر. هذا سيتسبب في رسالة ليتم عرضها في المربع التي نصها "This procedure was called from Object2".

```
Dim object1 As New Object1
Dim DelegateToInvoke As New Object2.DoStuff(AddressOf object1.DoStuff)
Dim object2 As New Object2(DelegateToInvoke)
```

وهذا مجرد مثال بسيط للكيفية التي يمكننا استخدام المندوبين لمساعدتنا على الخروج من حالات اللزجة ، واستخدام المندوبين هو بالتأكيد لا تقتصر على هذه الحالات.

الفصل الثامن عشر
المراجع

وفي بعض الوقت وربما نحتاج إلى أخرى تشمل الطبقة أو جمعيات المكتبات على طلبنا ، حتى أننا يمكن إن تشمل على وظيفة.

إشارة هي أساسا مكتبه المدونة التي نحن على صلة مشروعنا ، وكأنها من الدرجة المكتبة. وعندما نضيف إشارة ، يمكننا عندئذ استخدام أي كود داخل الإشارة إلى أن (والتي تتم العامة) لاحتياجاتنا.

مثال للاستخدام إشارة هو إضافة messengerapi المرجعية ، ولذا فإننا يمكن أن تدمج مع رسول أم أس إن ودردشة مفتوحة النوافذ ومستخدمي شبكة الانترنت للحصول على مركز من بين أمور أخرى.

وإذا أضفنا إلى المراجع مشروعنا ثم المجمع سوف نسخة هذه الإشارات لدينا الناتج مجلد المقبل لدينا تجميع ملف الجمعية ، وسوف نحتاج إلى أن تشمل هذه الإشارات حتى عندما ينفذ البرنامج انه يمكن الوصول إليهم.

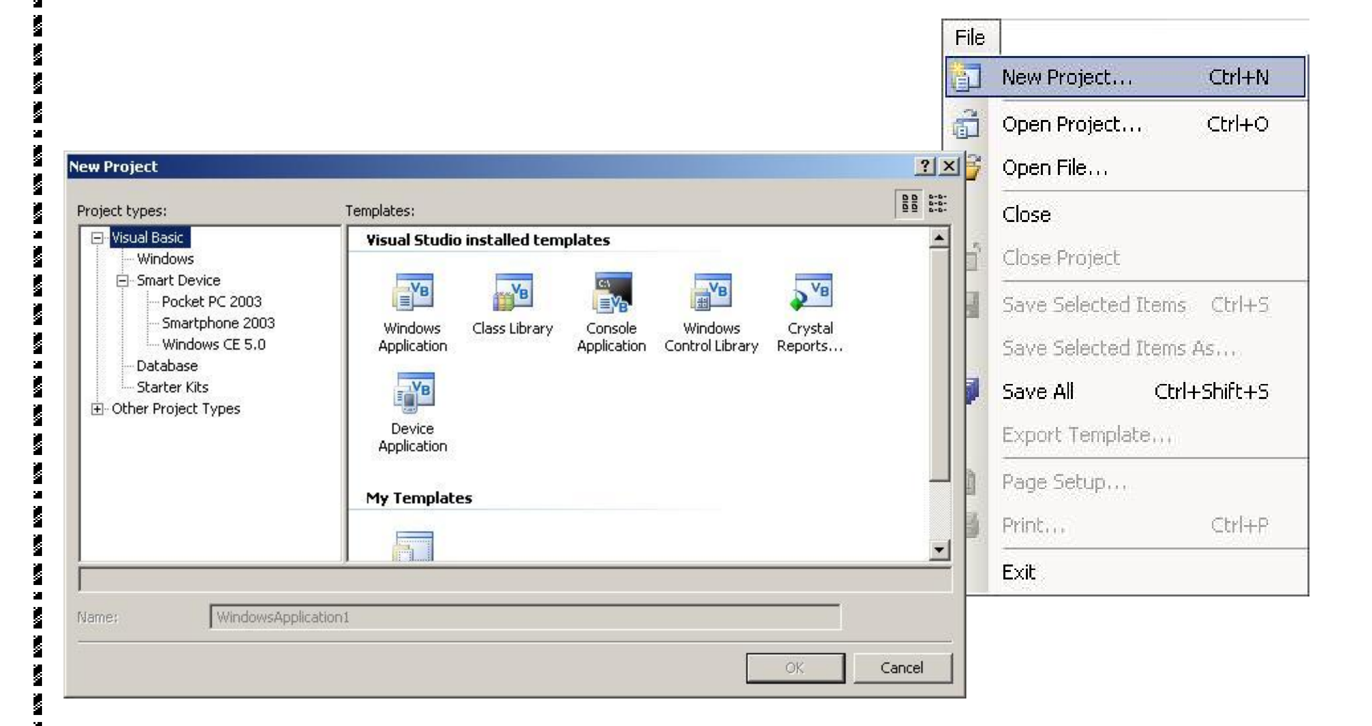
في.NET لدينا بالفعل ككل حمولة من الإشارات ، التي تسمى في إطار الطبقة المكتبة (fcl) ، ولكن هذه الملفات لن يحصل نسخت إلى الناتج مجلد ، لأنها شائعة في جميع آلات تشغيل. .NET Framework

الجزء الثاني
بيئة تطوير متكاملة

الفصل التاسع عشر

إنشاء مشروع جديد

في Visual Studio ومن السهل جدا إنشاء مشروع جديد. كل ما عليك فعله هو أن يفتح باب متابعة ، واختر File→New Project، وثم اختر نوع المشروع وسيتم خلق لكم.



الفصل العشرين

المشروع الممتلكات

لتغيير الخصائص المختلفة للمشروع الخاص بك ، انقر Project→<ProjectName> Properties

تطبيق علامة التبويب

يتيح لك تغيير مختلف البيئات التي تستخدم لتطبيق.

اسم الجمعية Assembly Name

الفعلي اسم الملف ستبنى ، وعلى سبيل المثال "windowsapplication1.exe" أو "windowsapplication1.dll".

نوع التطبيق Application Type

نوع من أن تطبيق سيتم تجميع.

الايقونه Icon

الايقونه التي سيتم عرضها عندما تقرأ تطبيق الجمعية(.EXE) في ويندوز اكسبلورر.

تجميع تبويبه Compile Tab

ويتضمن مختلف البيئات التي تتحكم في كيفية تطبيق هو تجميعها ، والمجلد حيث سيتم إنشاء الجمعية.

بناء الناتج الطريق Build Output Path

المجلد حيث سيتم تجميع الجمعية أنتج ل.

المراجع تبويبه References Tab

جميع القوائم الاضافيه مكتبات الدرجة المستخدمة في التطبيق ، ويتيح لك شيك معطله الإشارات.

الموارد تبويبه Resources Tab

ويوفر وسيلة لأنك بسهولة تضاف إلى الموارد التي يمكن استخدامها في طلبك ، على سبيل المثال الصور والنص الخيوط.

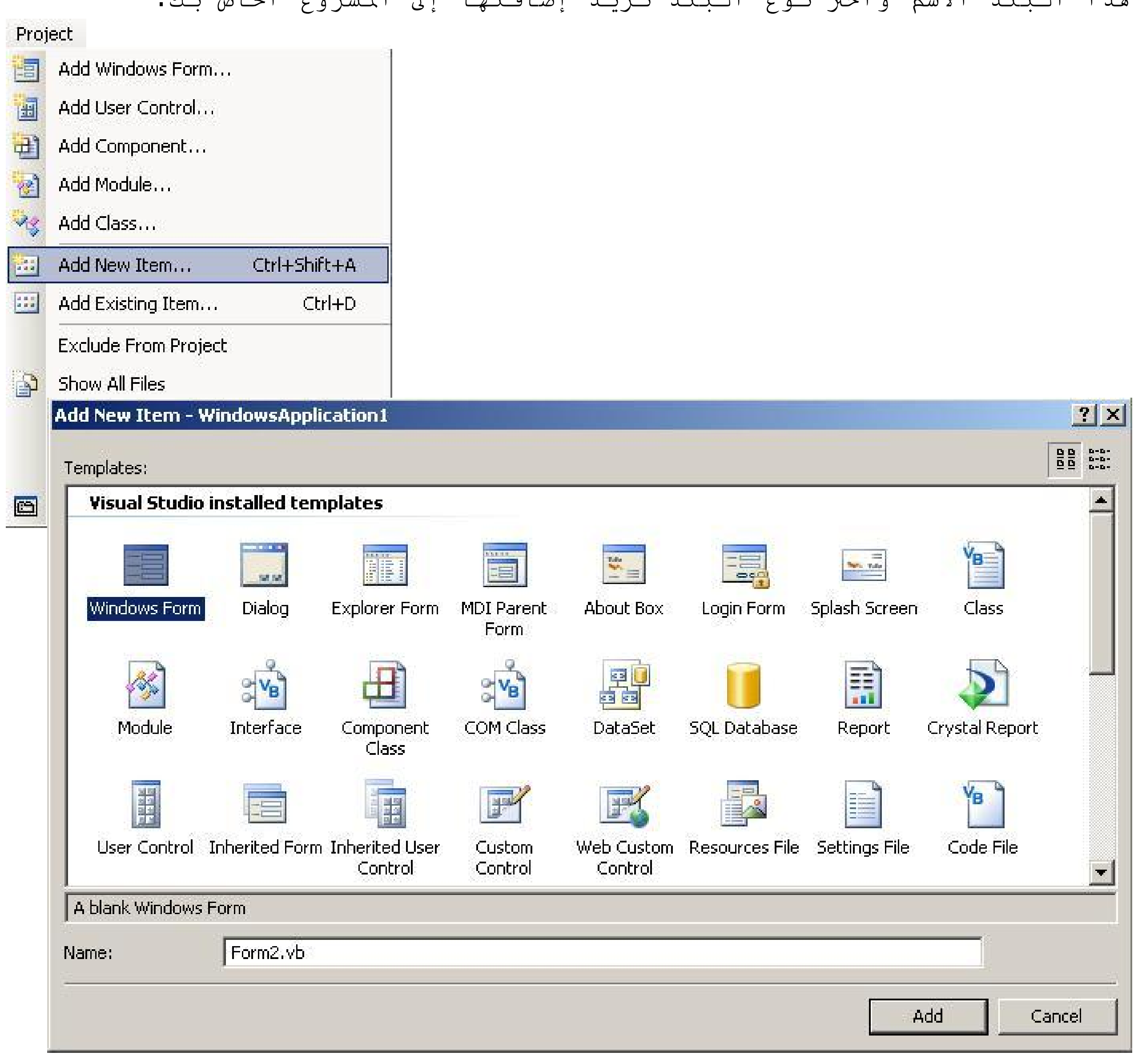
علامة تبويب الأعدادات Settings Tab

يتيح لك تخصيص أعدادات التي يمكن استخدامها في طلبك ، والذي أنقذ احصل على تطبيق إغلاق (افتراضي)

الفصل الحادي والعشرون

إضافة بنود جديدة

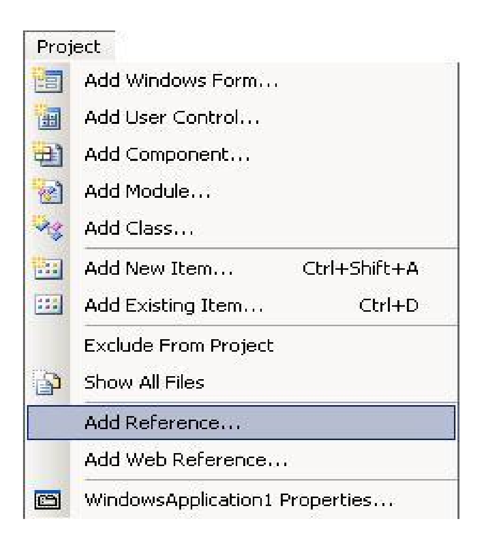
لإضافة بند جديد إلى طلبك من السهل جدا. انقر Project→Add New Item، وإعطاء هذا البند الاسم واختر نوع البند تريد إضافتها إلى المشروع الخاص بك.

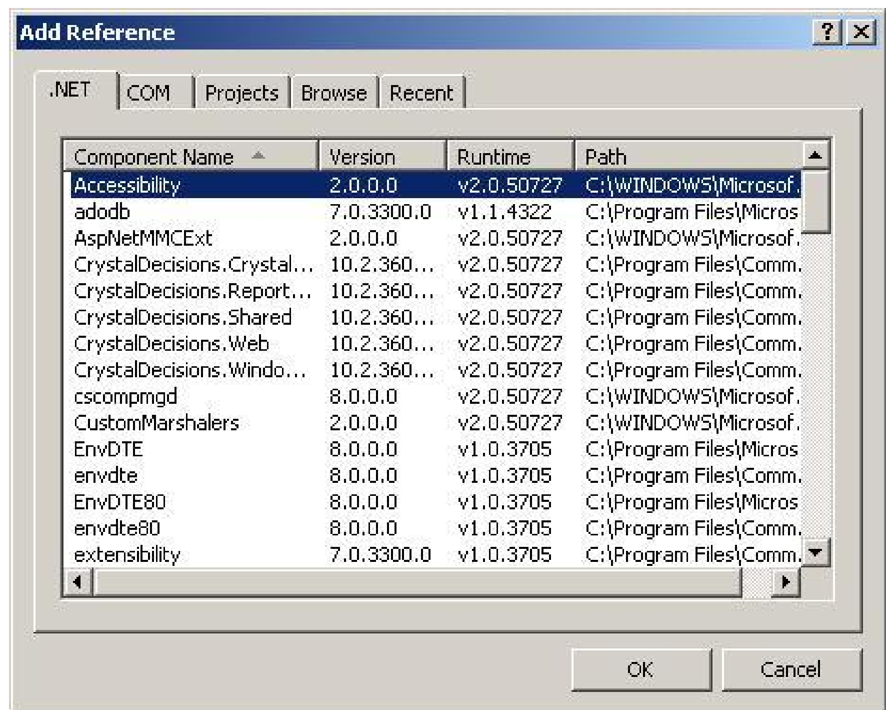


الفصل الثاني والعشرون

إضافة الإشارات

وربما نحتاج إلى إضافة أخرى جمعت المدونة على طلبنا في بعض الوقت. يمكننا أن نفعل ذلك بالنقر على Project→Add Reference، واختيار الجمعية إضافتها.

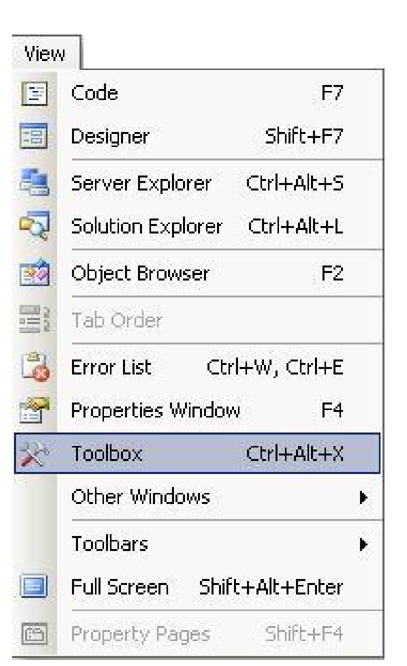


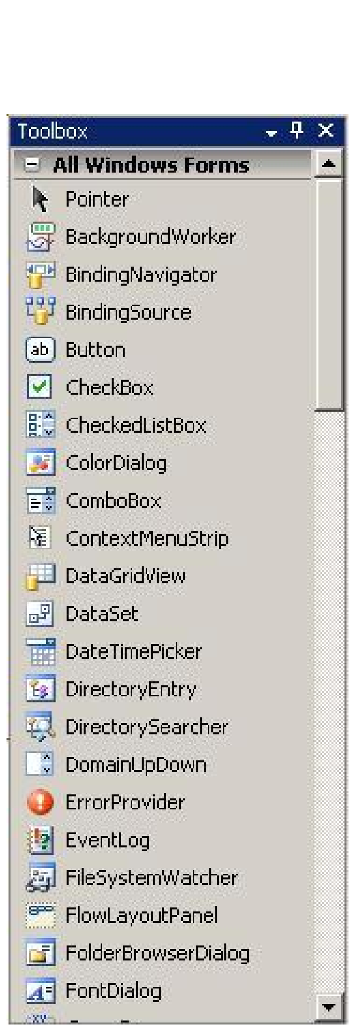


الـفـصـل الـثـالـث والـعـشـرون
صـنـدوق الأدوات

صندوق الأدوات تحتوي على قائمة من البنود التي يمكن إن نضيف إلى أشكال دينا ، على سبيل المثال الأزرار. لفتح صندوق الأدوات انقر على View→ToolBox.

يمكننا الآن أن اسحب بندا على شكل أعمالنا.

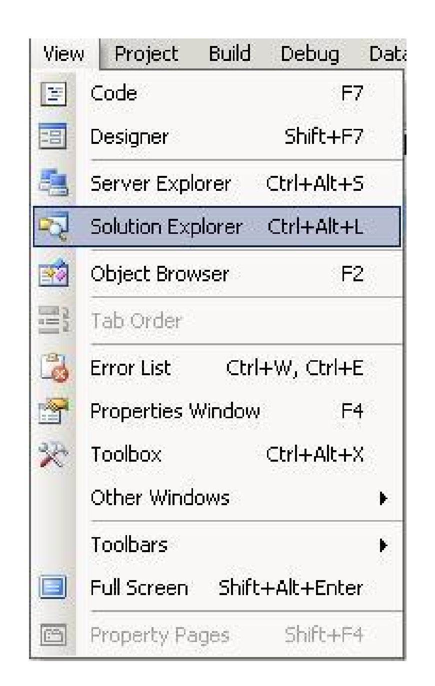


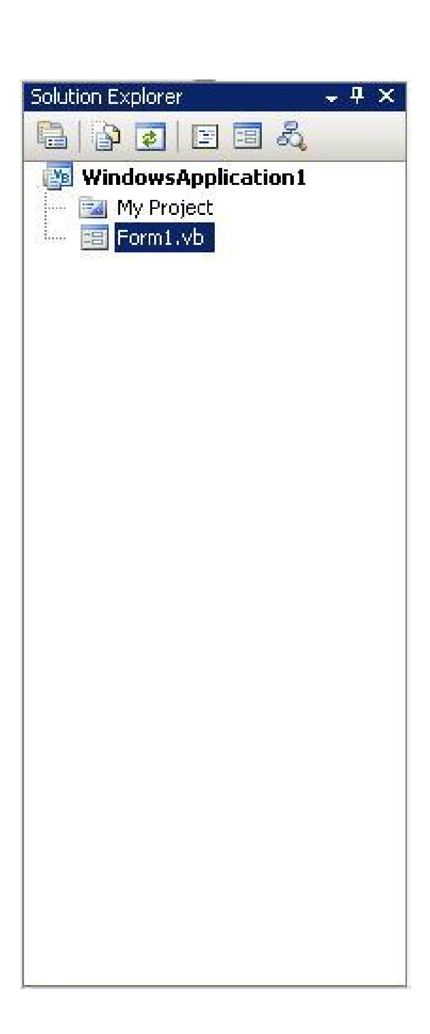


الفصل الرابع والعشرون
The Solution Explorer

The Solution Explorer قوائم جميع الملفات في مشروعنا ، ويوفر لنا طريقة سهلة للوصول إليها. فتح The Solution Explorer بالنقر View→Solution Explorer.

سترون في كل الطبقات ، الخ الإشكال المذكورة هنا.

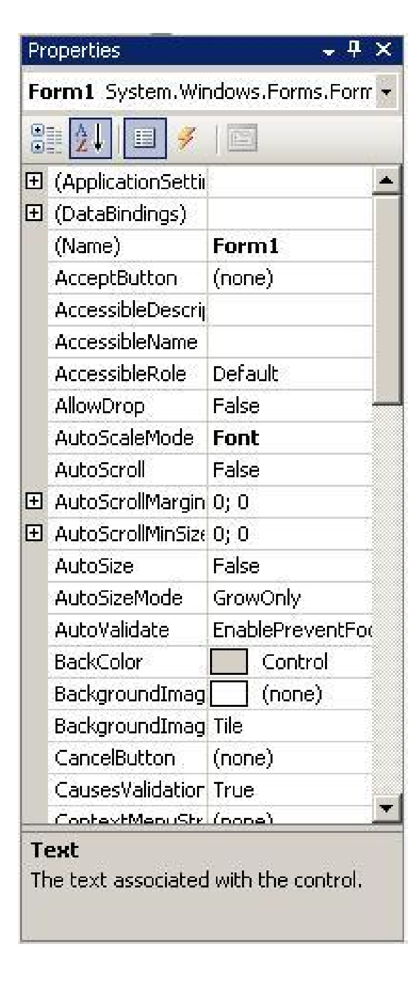




الفصل الخامس والعشرون
خواص النافذة

خواص نافذة تسمح لنا أن يتغير بسهولة الممتلكات بالنسبة للأجسام في بيئة تطوير متكاملة. تهيئة ويندوز المشروع ، حق انقر على form1 واختيار الممتلكات ، وعندئذ سترى خواص النافذة ، والذي سيحتوي على كل ما هو متاح من الممتلكات التي يمكنك تعديلها .

انقر على إسقاط القائمة في أعلى نافذة خواص لاختيار صنف (أو التي يجب أن تكون في نطاق) الذي تريد إن تعدل لممتلكات.



الجزء الثالث
انواع الادوات

الفصل السادس والعشرون
الضوابط المشتركة

Button

واحدة من انفع الضوابط الموجودة لدينا هي السيطرة على الزر ، وهو واضح.

Button1

الخصائص Properties

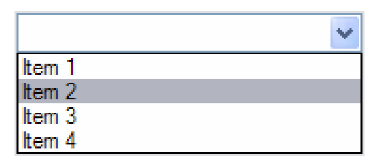
Text : النص يظهر على الزر.
Image : صورة التي تظهر على الزر.
Backcolor : لون الأزرار الخلفية.
Forecolor : لون الأزرار النص.

الإحداث Events

Click : يحدث عندما ينقر مستخدم على الزر ، او استخدام لوحة المفاتيح لاضغط على الزر.

صندوق منسدل ComboBox

Combobox هو إسقاط قائمة من البنود التي يمكن للمستعمل أن يختار منه.



الخصائص Properties

Autocompletecustomsource : مجموعة من الخيوط التي يمكن استخدامها لاستكمال السيارات.
Autocompletesource : المصدر أن الرقابة سوف تستخدم لإتمام السيارات ، على سبيل المثال "allurl" سيمكن السيارات الكامل لبنود في ويندوز موقع التاريخ للمستخدم الحالي.
Items : قائمة من البنود التي combo مربع يحتوي.
Sorted : تحديد ما إذا كان البنود في قائمة سيتم تلقائيا مرتبة.

الإحداث Events

DropDown : يحدث عندما إسقاط قطرات أسفل قائمة.
Selectedindexchanged : يحدث عندما يختار المستخدم مختلفة بند في القائمة.

العنوان Label

السيطرة الأساسية جدا ، ويستخدم لعرض النص على شكل من الإشكال.

Label1

الخصائص Properties

Text : النص الذي سيتم عرضها في العلامة.
Autosize : يمكن أو يعطل التلقائي ريسيزينج ، تبعا لحجم وإذا كان النص داخل السيطرة.
Textalign : مجموعات مواءمة النص في السيطرة.

Linklabel

أساسا نفس علامة السيطرة ، لكنه يعرض النص بنفس الطريقة التي ينشأ يمكن النقر على وصلة في صفحة ويب.

LinkLabel1

الخصائص Properties

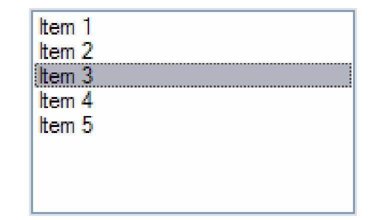
Text : النص الذي سيتم عرضها في العلامة.
Autosize : يمكن أو يعطل التلقائي ريسيزينج ، تبعا لحجم وإذا كان النص داخل السيطرة.
Textalign : مجموعات مواءمة النص في السيطرة.

الإحداث Events

Linkclicked : يحدث عندما ينقر مستخدم على الزر ، أو استخدام لوحة المفاتيح لأضغط على الزر.

Listbox

مراقبة بسيطة أن يعرض النص على قائمة البنود.



الخصائص Properties

Items : قائمة الأصناف المعروضة في السيطرة.
Borderstyle : نوع من الحدود إن تحصل على الانتباه للمراقبة.

الإحداث Events

Selectedindexchanged : يحدث عندما يختار المستخدم مختلفة البند في السيطرة.

Notifyicon

ويضيف أيقونه إلى الإخطار المجال ، وهو مجال على يسار الساعة.



الخصائص Properties

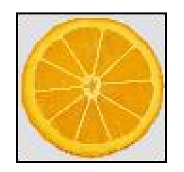
Text : النص الذي سوف يتم عرضها عند المستخدم الماوس يحوم حول السيطرة.
Icon : الايقونه التي سيتم عرضها في الأشعار المجال.
Visible : الجلود أو يظهر أيقونه في مجال الإخطار.
contextmenustrip : Contextmenustrip السيطرة التي سيتم عرضها عندما يقوم المستخدم بالنقر فوق الحق - أيقونه.

الإحداث Events

Mouseclick : ويحدث عندما يقوم المستخدم بالنقر فوق الايقونه.
Mousedoubleclick : يحدث عندما ينقر المستخدم مزدوجة الايقونه.

Picturebox

مراقبة الحاويات ، الذي يعرض صوره.



الخصائص Properties

Image : الصورة التي سيتم عرضها.
Imagelocation : موقع الصورة التي سيتم عرضها ، والتي يمكن أن يكون موقع ويب أو موقع على القرص.
Sizemode : الطريق إلى صورة داخل picturebox ستكون بحجم ثانية.

Progressbar

المستخدمة لعرض حالة عملية.



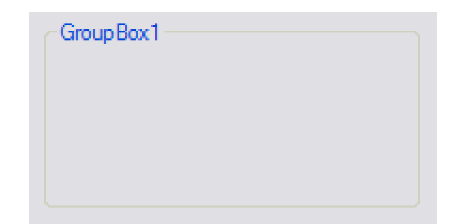
الخصائص Properties

Minimum : الحد الأدنى للقيمة.
Maximum : القيمة القصوى التي يمكن التوصل إليها.
Value : القيمة الحالية للprogressbar ، التي يجب أن تكون بين الحد الأدنى والحد الأقصى.
Style : طريقة progressbar سيتم استخلاصها ، إذا هو خيمة المختارة ومن ثم سوف تظهر باستمرار خيمة عبور progressbar ، الذي يستخدم عندما كنت لا تعرف كيف عملية طويلة ذاهب إلى اتخاذ.

الفصل السابع و العشرون
حاويات

Groupbox

ضوابط الجماعات الأخرى داخل علبة مع العلامة A.



الخصائص Properties

Text: النص يظهر في العلامة.

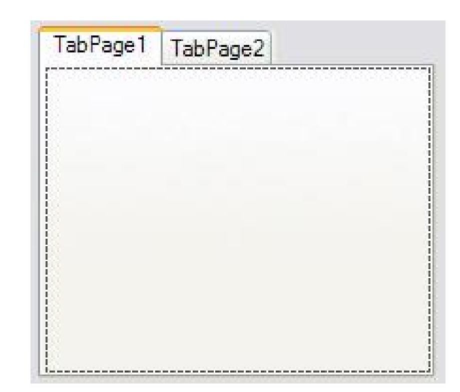
Panel

أساسا نفس groupbox ، ولكن من دون تسمية.



Tabcontrol

وتعقد tabpage الضوابط ، والتي يمكن أن تحتوي كل ضوابطها الخاصة.



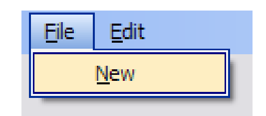
Properties الخصائص

Tabpages : مجموعة من tabpage الضوابط ، وكل منها يمكن أن يحتوي على غيرها من الضوابط.
MultiLine : يحدد ما إذا كانت خطوط متعددة من الأسعار يمكن أن تظهر في اعلي الصفحة .
Appearance : تغييرات النظرة للtabcontrol.

الـفـصـل الـثـامـن والـعـشـرون
الـقـوائـم واشـرطـه الأدوات

Menustrip

شريط من قائمة البنود ، التي هي شاءعه الاستعمال في معظم أشكال.



Properties الخصائص

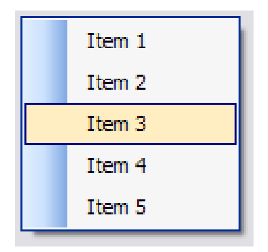
Items : مجموعة من toolstripmenuitem الضوابط ، التي تمثل جذور قائمة البنود التي ترد.

Events الإحداث

Itemclicked : ويحدث عندما يقوم المستخدم بالنقر فوق أي قائمة البند.

Contextmenustrip

تستخدم لإضافة السياق انقر بالزر الأيمن على قوائم المراقبة أو أشكال. أضفنا contextmenustrip إلى الجسم عن طريق تحديد دورته contextmenustrip إلى الممتلكات قيمة عادتنا contextmenustrip.



Properties الخصائص

Items : مجموعة من toolstripmenuitem الضوابط ، التي تمثل جذور قائمة البنود التي ترد.

Events الإحداث

Opening : قبل أن يتم فتح قائمة ، أو عرضها.
Opened : يحدث بعد قائمة فتح ، أو عرضها.
Closing : قبل أن يحدث هو قائمة مغلقة.
Closed : يحدث بعد هي قائمة مغلقة.

Statusstrip

مركز الشريط ، الذي يظهر عادة في أسفل تطبيق ويندوز ، والذي يمكن أن يحتوي على البنود المختلفة أداة الشريط.



الخصائص Properties

Items : مجموعة من toolstripitem الضوابط ، التي تمثل الضوابط التي تظهر على مركز الشريط.

Events الإحداث

Itemclicked : ويحدث عندما يقوم المستخدم بالنقر فوق احد بنود داخل السيطرة.

Toolstrip

شريط أدوات مثل مراقبة ، والذي يمكن أن يحتوي على العديد من مختلف toolstripitem الضوابط.



الخصائص Properties

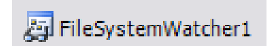
Items : مجموعة من toolstripitem ضوابط ، والضوابط التي تمثل الاداه التي تظهر على الشريط.

Events الإحداث

Itemclicked : ويحدث عندما يقوم المستخدم بالنقر فوق احد بنود داخل السيطرة.

الفصل التاسع والعشرون
المكونات

Filesystemwatcher



إن العناصر المكونة لنظام الساعات للملف الإحداث في المسار المحدد.

الخصائص Properties

Path : الدليل أو حملة لمشاهدة لملف الإحداث.
Enableraisingevents : يحدد ما إذا كانت filesystemwatcher تنشط أم لا.
Filter: ملف لرصد نمط للتغيرات.
Includesubdirectories : يقرر ما إذا كان ينبغي لرصد التغييرات في جميع أدلة ثانوية من المسار المحدد.
Notifyfilter : النوع من الملفات لرصد الأحداث.

الإحداث Events

Changed : يحدث عندما الملف في المسار المحدد معدلة.
Created: الملف يحدث عندما ينشأ في المسار المحدد.
Deleted : يحدث عندما الملف في المسار المحدد حذفها.
Renamed : يحدث عندما الملف في المسار المحدد هو اسمها.

Timer

الموقتات تستخدم لتشغيل مجموعة من البيانات المدونة على فترات محددة.



الخصائص Properties

Enabled : يقرر ما إذا كان ينبغي رفع انقضى وقت الإحداث.
Interval : مقدار الوقت الذي انقضى بين كل حال.

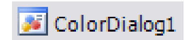
الإحداث Events

Elapsed : يحدث عندما الفترة الزمنية التي يحددها الفترة ينقضي. هذا هو أين نضع جهودنا المدونة التي نريد أن تدير بعد فترات.

الفصل الثلاثون
صناديق الحوار

Colordialog

ويسمح للمستخدم لاختيار لون ، بواسطة الحوار.

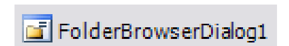


الخصائص Properties

Color : لون أنه عندما يتم اختيار الحوار هو مبين ، وكذلك يختار اللون المستخدم.

Folderbrowserdialog

ويتيح للمستخدم تحديد مجلد ، عن طريق الحوار.

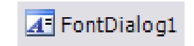


الخصائص Properties

Selectedpath : الطريق إلى إن الحوار يبدأ في عندما يتبين ، وأيضا الطريق إن المستخدم قد قام مختارة.

Fontdialog

ويتيح للمستخدم اختيار الخط ، عن طريق الحوار.



الخصائص Properties

Font : الخط وجوه أنه عندما يتم اختيار الحوار هو مبين ، وكذلك الخط وجوه أن المستخدم قد قام مختارة.

الإحداث Events

Apply: ويحدث عندما يقوم المستخدم بالنقر فوق تنطبق ، إلى "Apply" المختارة الخط.

Openfiledialog

ويتيح للمستخدم اختيار ملف ، أو ملفات متعددة ، لفتح ؛ عن طريق الحوار.



الخصائص Properties

FileName: ملف عندما يتم اختيار الحوار هو مبين ، وأيضا الملف إن المستخدم قد قام مختارة.
FileNames: ملف مجموعة من الأسماء أن المستخدم قد قام مختارة (إذا كان المستخدم قد قام نخبة من ملفات متعددة)
Filter : المرشح لنوع الملفات التي يمكن عرضها في الحوار.
Multiselect : تحدد ما إذا كان بامكان المستخدم اختيار عدة ملفات.
Title : العنوان هو إن يظهر في أعلى مربع الحوار.

Savefiledialog

ويسمح للمستخدم لاختيار مكان لحفظ الملف ، بواسطة الحوار.



الخصائص Properties

FileName: ملف عندما يتم اختيار الحوار هو مبين ، وأيضا الملف إن المستخدم قد قام مختارة.
Filter : المرشح لنوع الملفات التي يستطيع المستخدم باستثناء ما ورد.
Title : العنوان هو إن يظهر في أعلى مربع الحوار.

الجزء الرابع
مكتبات فصائل framework (FCL)

الفصل الحادي والثلاثون

system.drawing

العمل مع الرسامات

تحرير الصور

صورة الطبقة هي مفيدة جدا الطبقة ، والذي أنكم على الأرجح استخدام عند نقطة ما. وهو يتضمن أساليب لتحميل الصور ، تحويل الصور وحفظ الصور ، من بين أمور أخرى.

يمكننا بسهولة تحميل صورة من اسم الملف.

```
Dim image As Image
image = image.FromFile("C:\imageFile.bmp")
```

والآن أصبحت لدينا صورة وجوه ، والذي يمكننا إنقاذ لملف آخر.

```
image.Save("C:\newImageFile.bmp")
```

ويمكننا ايضا تحويل شكل الصورة إلى Jpeg ، على سبيل المثال.

```
Dim Format As Imaging.ImageFormat = Imaging.ImageFormat.Jpeg
image.Save("C:\newImageFile.bmp", Format)
```

إذا أردنا تحرير الصورة ، ونحن سوف يستخدم رسامات وجوه ، التي سنقوم ختامية حول صورة الجسم وثم استخدام الاعتماد على ذلك.

```
Dim graphics As Graphics = graphics.FromImage(image)
```

يمكننا الآن إن تعمل بشكل مباشر على الرسامات وجوه ، وسوف تعكس إي تعديلات على الصورة مباشرة.

بعض الرسامات وجوه يمكننا الاستفادة من أساليب لاستخلاص :

Graphics.drawarc : تلفت وهو قوس تمثل جزء من وجود البيضاوي يحددها مستطيل الهيكل.
Graphics.drawbezier : يستخلص bézier المفتاح التي حددها أربعة هياكل نقطة.
Graphics.drawbeziers : تلفت سلسلة من bézier المفاتيح من مجموعة من الهياكل نقطة.
Graphics.drawclosedcurve : تلفت مغلق الكاردينال المفتاح حددتها مجموعة من الهياكل نقطة.
Graphics.drawcurve : يستخلص الكاردينال المفتاح محدد من خلال مجموعة من الهياكل نقطة.
Graphics.drawellipse : يلفت احد البيضاوي يحددها لتنطط هيكل مستطيل.
Graphics.drawicon : تلفت الصورة التي يمثلها الرمز المحدد وجوه داخل منطقة يحددها مستطيل الهيكل.
Graphics.drawiconunstretched : تلفت الصورة التي يمثلها الرمز المحدد وجوه دون التوسع في الصورة.
Graphics.drawimage : تلفت صورة الجسم.
Graphics.drawimageunscaled : تلفت صورة الجسم دون التوسع.
Graphics.drawline : رسم الخط.
Graphics.drawlines : رسم سلسلة من الخطوط.
Graphics.drawpath : يستخلص graphicspath الجسم.
Graphics.drawpie : رسم شكل الفطيرة.
Graphics.drawpolygon : تلفت مضلع.
Graphics.drawrectangle : رسم مستطيل.
Graphics.drawrectangles : تلفت سلسلة من المستطيلات.
Graphics.drawstring : يستخلص النص الخيط.
Graphics.fillclosedcurve : تملأ داخلية مغلقة الكاردينال المفتاح المنحني.
Graphics.fillellipse : تملأ الداخلية من البيضاوي.
Graphics.fillpath : تملأ الداخلية لـلأ graphicspath الجسم.
Graphics.fillpie : تملأ الداخلية من الفطيرة الباب.
Graphics.fillpolygon : تملأ الداخلية للمضلع.
Graphics.fillrectangle : تملأ الداخلية للمستطيل.
Graphics.fillrectangles : تملأ الداخلية من سلسلة من المستطيلات.
Graphics.fillregion : تملأ المنطقة الداخلية من الجسم.

الفصل الثاني و الثلاثون
system.io

قراءة وكتابة النص ملفات

إذا أردنا إن المطلوب من أي وقت مضى إلى القراءة أو الكتابة ملفات النص ، وسنقوم بكل تأكيد استخدام streamreader وstreamwriter.

دعنا من قراءة الملف ثم نكتب البيانات التي تليت أمام ملف جديد.

إنشاء لجنة جديدة streamreader وجوه.

```
Dim streamReader As New IO.StreamReader("C:\file.txt")
```

خلق متغير لتحميل البيانات.

```
Dim text As String
```

قراءة البيانات.

```
text = streamReader.ReadToEnd()
```

إغلاق هذا الملف.

```
streamReader.Close()
```

إيجاد streamwriter لكتابة البيانات.

```
Dim streamWriter As New IO.StreamWriter("C:\newFile.txt")
```

أكتب البيانات إلى الملف.

```
streamWriter.Write(text)
```

إغلاق streamwriter الأمر الذي يفرض على جميع البيانات الذي لم يدخل بعد المكتوبة إن flushed إلى الملف.

```
streamWriter.Close()
```

الجزء الخامس
الشفرات

الفصل الثالث والثلاثون
مرحبا العالم!

إنشاء مشروع جديد

أولا وقبل كل شيء ، وتفتح Visual Basic 2005. نحن نذهب إلى خلق جديد ويندوز حتى انقر File→New Project، اختر "Windows Application" ، في نوع " Hello World!"!" بوصفها اسم والصحافة Ok.

أنت الآن قدمت مع ويندوز أشكال مصمم.

نوافذ أشكال مصمم

نوافذ أشكال مصمم أساسا وسيلة لتحرير النظرة من أشكال إعمالنا ، مما يجعل الحياة أسهل بكثير بالنسبة لنا.

يمكننا أن اسحب البنود على شكل ، مثل الأزرار والقوائم ، وباستخدام مجموعة ممتلكاتهم خواص النافذة. أشكال مصمم يولد. الصافية المدونة (Visual Basic المدونة في هذه القضية (هل فعلا أن إنشاء شكل وعلى كل واحد من وجوه أنها بالنسبة لنا. الجمال من هذا هو انه عندما خلق شكل وإضافة بضع تعترض عليه ، وهناك مئات من الخطوط المدونة انه لم يعد لدينا لنوع!

مضيفا لدينا شكل الأجسام

دعونا نضيف بعض الأشياء لدينا الآن شكل. فتح صندوق العدة به انقر على عرض / صندوق العدة أو تحوم فوق القليل مربع على اليسار إن يقول صندوق العدة.

وهناك فئات كثيرة من الأجسام يمكننا الآن أن نضيف لدينا أشكال.

ولندع إضافة زر التحكم وlistbox السيطرة. انقر الضوابط المشتركة ، وأسحب زر واحد listbox الجسم على شكل أعمالنا. اسحب منهم حوالي حتى أنت مريحة مع مواقعها.

لديك ربما لاحظ أن الزر يقول "button1". أن تفتح نافذة على الممتلكات الحق مثلما فعلتم مع صندوق العدة (أو الصحافة f4) و من ثم تظهر القائمة المنسدله في أعلى اختر button1. الآن يمكنك أن تحدد الممتلكات لbutton1. حدد النص على أنها "My Button" أو أيا كانت تريد. أيضا اختيار form1 من قائمة التغيير ونصه إلى "Hello World!".

ensert.net2000@gmail.com www.Ensertnet2000.all-forum.net

مضيفا مدونة لدينا الأجسام

طيب ، ونحن الآن مستعدون ليضفي مسحة من مدونة لطلبنا. ما سنفعله هو أن تضيف دخول إلى listbox في كل مرة ينقر المستخدم على الزر. سنكون باستخدام button.click الحدث ليخبرنا انه ينبغي أن نضيف دخول. الحق انقر في اي مكان في أشكال المصمم وانقر على عرض المدونة. ونحن الآن في تقديمها مع :

```
Public Class Form1

End Class
```

هذا ، كما يمكنك أن تعرف جيدا ، هي الطبقة إعلان لform1.

أضف حدثا لbutton.click خلال فتح إسقاط قائمة على أعلى اليسار للمدونة المجال واختيار button1 ، ثم فتح إسقاط قائمة على الحق واختيار انقر.

ونحن ألان نظرا للمدونة انقر الحدث.

```
Private Sub Button1_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button1.Click
End Sub
```

إشعار بأن تتولى هي الإشارة إلى button1.click الحدث.

سنقوم الآن أن أضيف بعض المدونة بحيث أننا في كل مرة انقر على زر دخول جديدة تحصل إضافته إلى قائمة المربع في شكل "Entry Number: #" ، حيث # وعدد من دخول المضافة.
لذلك نحن نذهب إلى إضافة متغير الخاص ، ودعا entrynumber في إعلان لform1.

```
Private EntryNumber As Integer = 0
```

أحب أن أضيف إلى بلادي قبل كل شيء متغيرات الحدث وطريقة الإعلانات ، ولكن يمكنك أن تضيف هذا أينما تشاء (يمكن قبل الحدث أو بعده الإعلان ، ولكن يجب أن تكون داخل form1 الإعلان).

كل مرة ينقر مستخدم ، ونحن سوف الزيادة entrynumber جانب واحد ، خلق النص شريطا لتخزين نص بأننا سوف تضيف إلى قائمة مربع ، يضاف إليها النص وإرفاق entrynumber إليها هاتفيا tostring الطريقة التي المتحولون إلى ا الخيط ، وبعد ذلك يضيف النص إلى قائمة المربع.
نحن المدونة داخل الحدث إعلان لbutton.click ينبغي أن ننظر مثل هذا الشيء :

```
'Increment EntryNumber.
Me.EntryNumber += 1

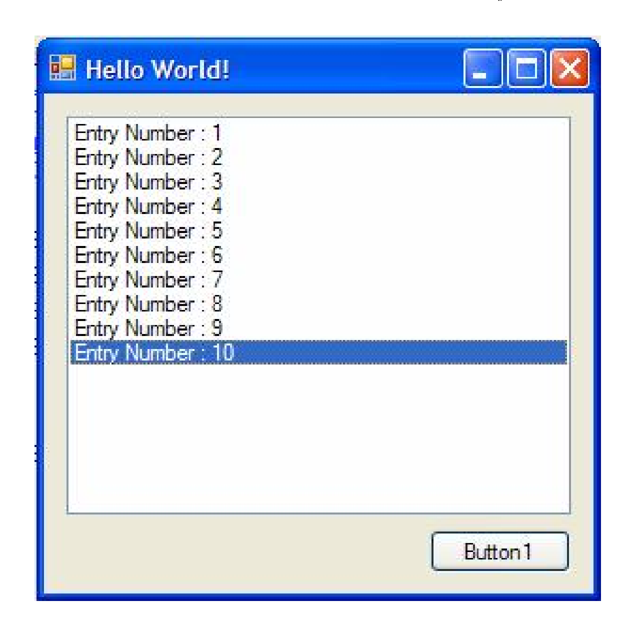
'Store a text string for the entry we will add.
Dim textEntry As String = "Entry Number : " & Me.EntryNumber.ToString

'Add it to the listbox.
Me.ListBox1.Items.Add(textEntry)
```

اختبار طلبنا

إدارة المشروع عن طريق النقر Debug→Start Debugging والاختبار بها بالنقر على الزر بضع مرات.

وينبغي لنا أولا تطبيق هذا أشبه :



شفره المصدر الكامل لهذا المشروع

```
Public Class Form1

    Private EntryNumber As Integer = 0

    Private Sub Button1_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button1.Click
        'Increment EntryNumber.
        Me.EntryNumber += 1

        'Store a text string for the entry we will add.
        Dim textEntry As String = "Entry Number : " &
Me.EntryNumber.ToString

        'Add it to the listbox.
        Me.ListBox1.Items.Add(textEntry)
    End Sub

End Class
```

الفصل الرابع الثلاثون
إنشاء Magic 8 Ball

ما هو سحرى ٨ الكرة؟

سحري ٨ الكرة لأول مرة اخترعت في si'٦٠ نعتقد. ومن أساسا إضافيا حوض كبير الكرة ، مع ثقب في أعلى التي تظهر إجابات مختلفة في كل مرة تقوم فيها هزة.

تصميم سحرى ٨ الكرة

دعونا نفكر في بعض الاحتياجات لمشروعنا.

١. وسوف يكون الشكل الأساسي ، مع زر أن عروض جديدة في كل مرة أنها نتيجة للضغط.
٢ - سيكون لدينا ثمانية الكرة في هذا المركز ، الذي هو الانتباه على آ picturebox السيطرة. عندما شكل الاحتمال أي نتيجة سيتم عرضها ، وسوف يكون مجرد فراغ.
٣. سنقوم تخزين أكثر من جمع الخيوط لمختلف النتائج الممكنة وتنتقي منها بشكل عشوائي في كل مرة هو النقر على الزر. سيكون لدينا نتائج والعشرين ، وكأنها طبيعية سحريه ٨ الكرة.

مضيفا لدينا شكل الأجسام

ونحن أول من سوف تضاف picturebox السيطرة. اسحب واحد ونحن على شكل وحجم المجموعة (في نافذة الخصائص) إلى ٢٥٠٠٢٥٠. في الوسط picturebox أفقيا (Format→ Center In Form→Horizontally). تغيير شكل قليلا لتتناسب مع انه في ذلك الحين وإضافة زر المربع أدناه الصورة. مجموعة الزر النص إلى "Ask a Question" ، لأنها تناسب تغيير في النص إذا اقتضى الأمر ، وأنها مركز أفقيا. مجموعة أشكال النص إلى "Magic 8 Ball" و backcolor إلى الأبيض ، ويمكننا أن ننتقل إلى الترميز.

مضيفا مدونة لدينا الأجسام

أولا وقبل كل شيء ، ولدينا إعداد الردود العشوائيه. نحن نذهب إلى متجر لها في stringcollection ، وهي مجموعة خاصة من نوع أن سلسلة مخازن القيم. تضاف عليه داخل form1 للإعلان الطبقة.

```
Private RandomResponses As Collections.Specialized.StringCollection
```

ولجعل إجراءات لتسكين هذه المجموعة مع استجابة عشاءيه الخيوط ، initrandomresponses.

```
Sub InitRandomResponses()

End Sub
```

يضاف ما يلي المدونة داخل هذا الفرعية.

```
Me.RandomResponses = New Collections.Specialized.StringCollection
Me.RandomResponses.Add("Signs point to yes.")
Me.RandomResponses.Add("Yes.")
Me.RandomResponses.Add("Reply hazy, try again.")
Me.RandomResponses.Add("Without a doubt.")
Me.RandomResponses.Add("My sources say no.")
Me.RandomResponses.Add("As I see it, yes.")
Me.RandomResponses.Add("You may rely on it.")
Me.RandomResponses.Add("Concentrate and ask again.")
Me.RandomResponses.Add("Outlook not so good.")
Me.RandomResponses.Add("It is decidedly so.")
Me.RandomResponses.Add("Better not tell you now.")
Me.RandomResponses.Add("Very doubtful.")
Me.RandomResponses.Add("Yes - definitely.")
Me.RandomResponses.Add("It is certain.")
Me.RandomResponses.Add("Cannot predict now.")
Me.RandomResponses.Add("Most likely.")
Me.RandomResponses.Add("Ask again later.")
Me.RandomResponses.Add("My reply is no.")
Me.RandomResponses.Add("Outlook good.")
Me.RandomResponses.Add("Don't count on it.")
```

الآن نحن بحاجة إلى إيجاد وظيفة من أن العائدين عشاءيه استجابة من هذه المجموعة. سنطلق عليه getrandomresponse ، وانه سيعود سلسلة القيمة تتضمن استجابة عشاءيه.

```
Function GetRandomResponse() As String

End Function
```

يضاف ما يلي المدونة داخل هذه الوظيفة.

```
'Create a new Random object.
Dim random As New Random

'Get the highest response index possible.
Dim highestResponse As Integer = Me.RandomResponses.Count

'Use Random.Next to get a random value between 0 and highestResponse.
Dim responseIndex As Integer = random.Next(0, highestResponse)

'Return the random response, using the index we just got.
Return Me.RandomResponses(responseIndex)
```

ونحن الآن ستضيف الأجراء أن تلفت صورة سوداء مع الكرة والنص ، على أن يوضع في منطقتنا صورة المربع. سنطلق عليه updateimage ، وهذا سيكون له رد بارامتر من النوع الخيط الذي سوف نستخدم لتمرير عشاءيه ردا على ذلك انه ينبغي أن تستفيد.

```
Sub UpdateImage(Response As String)

End Sub
```

يضاف ما يلي المدونة داخل هذا الفرعية.

```
'Create a new image object
Dim magic8Ball As New Bitmap(250, 250)

'Create a graphics object to draw on the image.
Dim graphicsObject As Graphics = Graphics.FromImage(magic8Ball)

'Set the SmoothingMode, for smoother antialiased drawing.
graphicsObject.SmoothingMode = Drawing2D.SmoothingMode.AntiAlias

'Create a rectangle, which will be used to draw the ellipse.
Dim circleRectangle As New Rectangle(0, 0, 250, 250)

'Draw the ellipse.
graphicsObject.FillEllipse(Brushes.Black, circleRectangle)

'Create a rectangle to draw the text string in.
Dim textRectangle As New Rectangle(0, 0, 250, 250)

'Create a StringFormat object to setup the text alignment to center.
Dim stringFormat As New StringFormat
stringFormat.Alignment = StringAlignment.Center
stringFormat.LineAlignment = StringAlignment.Center

'Create a font to draw the response with.
Dim font As New Font("Comic Sans MS", 10, FontStyle.Bold)

'Draw the response string.
graphicsObject.DrawString(Response, font, Brushes.White, textRectangle,
stringFormat)

'Cleanup the graphics object, to save memory.
graphicsObject.Dispose()

'Set PictureBox1.Image to the image we created.
Me.PictureBox1.Image = magic8Ball
```

ونحن الآن بحاجة إلى إضافة بعض المدونة لform1.load الحدث حتى يمكننا أن ندعو initrandomresponses واستخلاص بياض ٨ الكرة. انقر على إسقاط القائمة في أعلى اليسار ، واختر form1 الإحداث ، ثم اختر تحميل من قائمة على أعلى يمين الصفحة. يضاف ما يلي المدونة داخل هذا الحدث.

```
'Initialize random responses.
Me.InitRandomResponses()

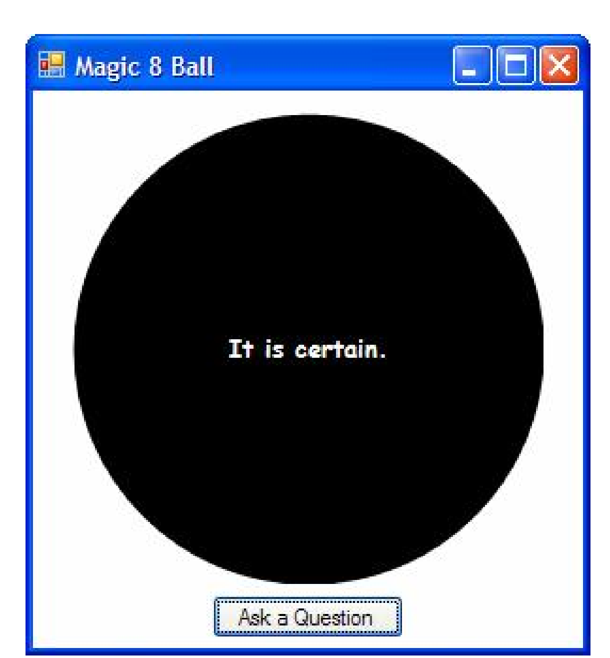
'Update the image, with no text.
Me.UpdateImage(Nothing)
```

وأخيرا ، أننا بحاجة إلى إضافة المدونة لbutton1.click الحدث أن ما سيحصل عليه عشوائيا واستجابة الدعوة updateimage مع استجابة عشاءيه. الوصول button1.click الحدث مثلك فعلت مع form1.load الحدث أعلاه ، ثم تضيف الشفرة التالية.

```
'Get a random response and update the image.
Me.UpdateImage(Me.GetRandomResponse)
```

اختبار طلبنا

وأخيرا يمكننا اختبار لنا ماجيك ٨ الكرة. سؤال أطرحه ، ثم تضغط على زر وستحصلون على رد في نص أبيض في خط الوسط. عندما يركض ، وينبغي أن ينظر مثل هذا الشيء :



الفهرس

الجزء الثالث
أنواع الأدوات

الجزء الرابع
مكتبات فصائل framework (FCL)

الجزء الخامس
الشفرات

المحتويات